User Guide

Book a Call - info@raven-apps.com - Package Version - FAQs - Pricing - AppExchange

Introduction

Introducing Grids by RavenApps for Salesforce on AppExchange
How to Boost Salesforce Adoption & Productivity with RavenApps for Sales, Service, Marketing &B…

Our complement ‘**Grids > by RavenApps**’ can be added via Lighting App Builder to lightning pages such as Record Page, App Page and Home Page.

The component allows you to show any records and any fields on a page from any part of the Salesforce database, i.e. not restricted to child or parent relationship records.

Read More
Where can you place grids?
Grid Builder

Installation

1. Install Package

1.1. Go to the Salesforce AppExchange Grids page by clicking here

1.2. Click ‘Get It Now’

1.3. Choose to Install in Production or in Sandbox

1.4. Agree to the T&Cs and click ‘Confirm and Install’, you will be prompted to provide the credentials for this org.

1.5. Choose to ‘Install for Admins’ then click ‘Install’

Note: See Package Releases for installation of latest version, release notes, coming soon.

2. Add User License & Permission Set

2.1. In Setup under 'Installed Packages' find the Grids package, click 'Manage Licenses' and add users as required.

2.2. See Permission Set section for optionally adding this for users.

3. Go to App

Click on the 9 dot waffle button on the top left of the screen and search for the 'Grids' application. (see Quick Start & Tips).

Updates

We are continuing to update the app with new Package Versions -see this link for installation of the latest version, release notes, coming soon.

There are sometimes advantages to installing the app directly from a url rather than having to log into AppExchange, to do this see the details here.

The url to use is below and you will need to update My Domain:
https://MyDomainName.lightning.force.com/packagingSetupUI/ipLanding.app?apvId=...

You can find the latest version by following the trick here or by emailing info@raven-apps.com

Quick Start & Tips

1 minute video here running through the below 2 steps for a Related List Grid.

1. Create a Grid
2. Add Grid to a Lightning Page
3. Tips

1. Create the Grid Record

Go to the 'Grids' Object and click 'New'. You can create either a Related List Grid (to be put on a record page) or an Explorer Grid (a stand alone grid).

Read More: Grid Actions

2. Add Grid to Lightning Page

On Lightning App Builder drag from left hand Components side the 'Grids > by RavenApps' to where to desire on the Lightning App page.

With the Grid component added and in focus, from the right hand component details set the Grid Id. This can be compiled an pasted from the Grid record. We recommend update the auto populated ‘Grid Id (UID__c)’ with readable friendly name e.g. Open Opportunities

Optionally you can also set the visibility settings of the component and link to the Custom Permission / Permissions Set.

3. Tips

Above steps is just the beginning of initializing the Grid; there are lots more capabilities and features which can be added.

Check out our CRM Grids tab -> Open Opportunities which shows some other capabilities in action if you have Opportunity records in your system.

See quick video below showcasing some of our key features:
How to Boost Salesforce Adoption & Productivity with RavenApps for Sales, Service, Marketing &Beyond

Copy and Paste from our provided Open Opportunities grid record for Column, Filter, Sort, … definitions which you can then adjust specifics based on your context.

Do Book a Call or Contact Us if you are running into questions or issues.

Features

There are a number of areas that can be enhanced in a grid after it has been created using the grid builder:

Parent Fields
Clone Row
Column Settings
Completion Bar Columns
Conditional Formatting
Default values for inline create
Download
Dynamic Grids
Field Header Icons

Filters
Image Cells
Invoke a Flow
Related Grids
Sorts
Sparklines
Summaries
Views
Views - Show Last Viewed Grid

Column Types

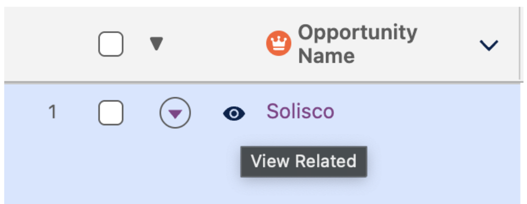
We utilize a number of standard field UI based on Salesforce Datatable and others we have custom created.

#	Column	Standard	Custom
1	View Related		✓
2	Name		✓
3	RecordPicker (aka Lookup / Master Details)		✓
4	Picklist		✓
5	Multi-Picklist		✓
6	Long Text		✓
7	Rich Text		✓
8	Click-to-Dial		✓
9	Formula Image E.g. Photo, Completion bar, Sparkline		✓

View Related

Last Update 13th May 2025

This column allows for a per row clickable icon to open a pop over modal to show related grids. Like on Open Opportunities grid drill into see related Contact Roles, Notes, Event, Tasks while staying on the same page.

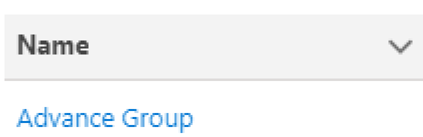


{"label":"","fieldName":"related","value":"Related","type":"button-icon","iconName":null,"fixedWidth":25,"editable":false,"sortable":false,"hideLabel":true, "typeAttributes":{"name":"ra_viewRelated","iconName":"utility:preview","alternativeText":"View Related","variant":"bare"}},

Name

Last Update 13th May 2025

This allows a record name with hyperlink to used for a Grid cell and uses the field type of *ra_name*:

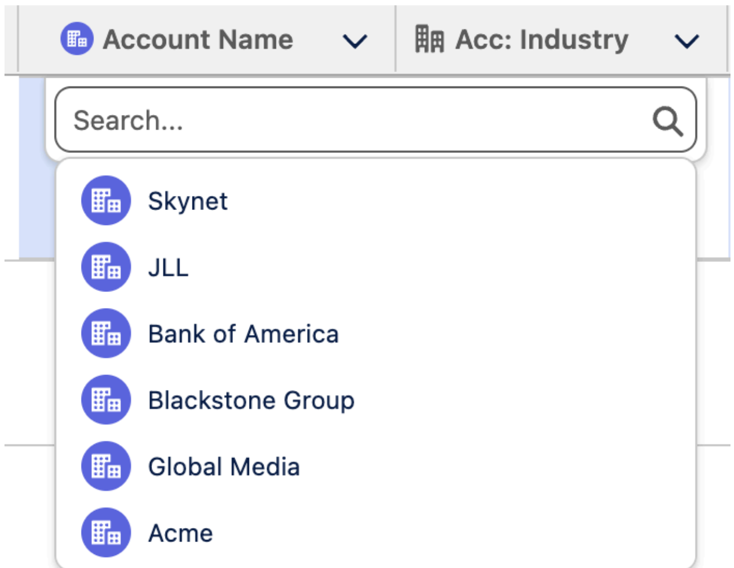


An example JSON column for this would be for the Account name field on Account object:

{"label":"Opportunity Name", "fieldName":"record.Name", "value":"Name", "type":"ra_name2", "sortable":true, "editable":true, "initialWidth":170, "iconName": "standard:opportunity", "wrapText":true, "typeAttributes": {"navRecordId": { "fieldName": "record.Id" }, "target": "_self" }},

RecordPicker (aka Lookup / Master Detail)

Last Update 3rd June 2025



Example

{"label":"Account Name", "value":"AccountId","fieldName":"record.AccountId", "editable":true,"iconName":"standard:account","initialWidth":170,"sortable":true,"type":"ra_recordPickerC", "wrapText":true, "typeAttributes":{"valueLabel":{"fieldName":"record.Account.Name"}, "valueLabelField":"Name", "object": "Account", "subTitleField":null, "whereFilter":null, "whereStringVal1":null, "whereIntegerVal1":null, "target":"_self", "icon": "standard:account"}},

Filtered Record Picker Examples (Paid Version)

Hard Coded

{"label":"Account Name", "value":"AccountId","fieldName":"record.AccountId", "editable":true,"iconName":"standard:account","initialWidth":170,"sortable":true,"type":"ra_recordPickerC", "wrapText":true, "typeAttributes":{"valueLabel":{"fieldName":"record.Account.Name"}, "valueLabelField":"Name", "object": "Account", "subTitleField":null, "whereFilter": "Type IN ('Customer - Direct’, 'Customer - Channel')", "whereStringVal1":null, "whereIntegerVal1":null, "target":"_self", "icon": "standard:account"}},

String Variable

{"label":"Contact Name", "value":"ContactId","fieldName":"record.ContactId", "editable":true,"iconName":"standard:account","initialWidth":170,"sortable":true,"type":"ra_recordPickerC", "wrapText":true, "typeAttributes":{"valueLabel":{"fieldName":"record.Contact.Name"}, "valueLabelField":"Name", "object": "Account", "subTitleField":”Account.Name”, "whereFilter": "AccountId =: whereStringVal1", "whereStringVal1":{"fieldName": "record.AccountId"}, "whereIntegerVal1":null, "target":"_self", "icon": "standard:account"}},

Integer Variable

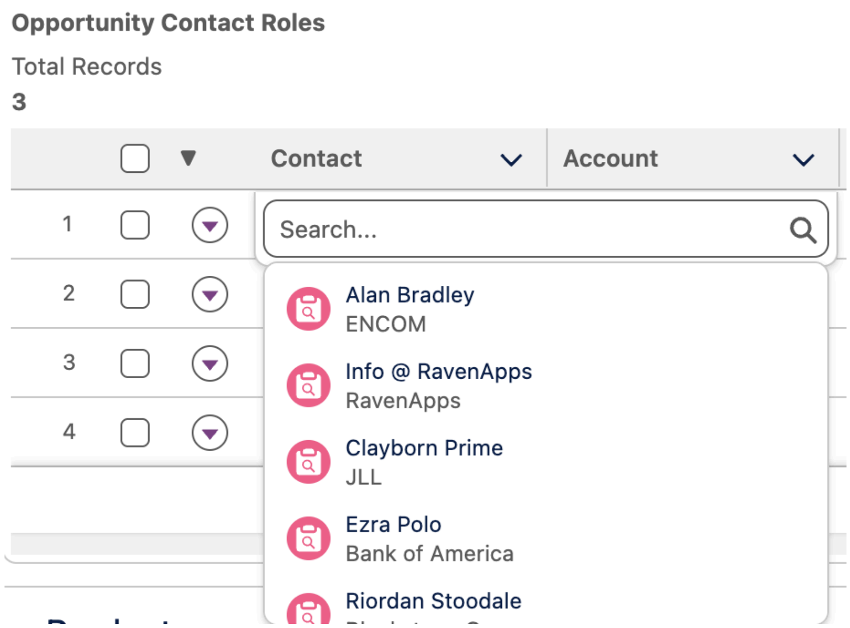
{"label":"Contact Name", "value":"ContactId","fieldName":"record.ContactId", "editable":true,"iconName":"standard:account","initialWidth":170,"sortable":true,"type":"ra_recordPickerC", "wrapText":true, "typeAttributes":{"valueLabel":{"fieldName":"record.Contact.Name"}, "valueLabelField":"Name", "object": "Account", "subTitleField":"Account.Name", "whereFilter": "Account.AnnualRevenue >= :whereIntegerVal1", "whereStringVal1":null, "whereIntegerVal1":100, "target":"_self", "icon": "standard:account"}},

Exclude Current Record (e.g. for Parent Account)

{"label":"Parent", "value":"ParentId","fieldName":"record.ParentId", "editable":true,"iconName":"standard:account","initialWidth":170,"sortable":true,"type":"ra_recordPickerC", "wrapText":true, "typeAttributes":{"valueLabel":{"fieldName":"record.Account.Name"}, "valueLabelField":"Name", "object": "Account", "subTitleField":null, "whereFilter": "Id != :rowId", "whereStringVal1":null, "whereIntegerVal1":null, "target":"_self", "icon": "standard:account"}},

Known Limitation: For inline creation if the variable is referring to the current record being created the filter will not have a value to refer to and hence will not find any values. The work around for this is to either save the record and then update this one field or do not enforce the filter if inline creation is needed.
Alternatively can use a flow for record creation.

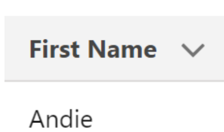
Subtile



{"label":"Contact", "fieldName":"record.ContactId", "value":"ContactId", "type":"ra_recordPickerC", "editable":true, "sortable":true, "iconName":null, "initialWidth":null, "typeAttributes":{"valueLabel":{"fieldName":"record.Contact.Name"}, "valueLabelField":"Name", "object":"Contact", "subTitleField":"Account.Name", "whereFilter":null, "whereStringVal1":null, "whereIntegerVal1":null, "target":"_self", "icon":"standard:record_lookup" } },

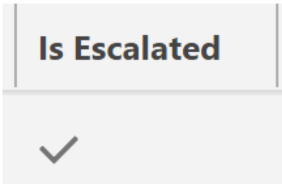
Text

This field type uses the type of “text”. Example JSON:
{"label":"First Name","fieldName":"record.FirstName","value":"FirstName","type":"text","sortable":true,"editable":true, "initialWidth":100}



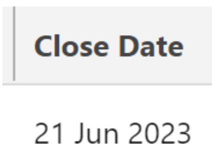
Boolean

This field type uses the type of “boolean”. Example JSON:
{"label":"Is Escalated","fieldName":"record.IsEscalated","value":"IsEscalated","type":"boolean","sortable":true,"editable":true, "initialWidth": 100}



Date

This field type uses the type of “date-local”. Example JSON:
{"label":"Close Date", "fieldName":"record.CloseDate", "value":"CloseDate", "type":"date-local", "editable":true, "sortable":true, "initialWidth": 200}

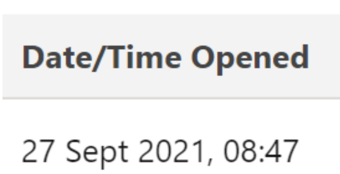


Date Time

This field type uses the type of “date” and is also added in the typeAttribute below.
"typeAttributes": {"year": "numeric","month": "short","day": "2-digit","hour": "2-digit","minute": "2-digit"}

Example 1
{"label":"Date/Time Opened", "fieldName":"record.CreatedDate", "value":"CreatedDate", "type":"date", "editable":true, "sortable":true, "initialWidth": 150, "typeAttributes": {"year": "numeric","month": "short","day": "2-digit","hour": "2-digit","minute": "2-digit"}}

The type attribute JSON in the above is used so that the view screen displays the date time values:



Without the attribute section the view screen would have just the date and the edit screen would have date and time:

Customizing date formats

The date / date time format is drive by 2 area:

1. The locale of the end users
2. Include in the column JSON:
    a. Datetime fields: "typeAttributes": {"year": "numeric","month": "short","day": "2-digit","hour": "2-digit","minute": "2-digit"}
    b. Date fields: "typeAttributes": {"year": "numeric","month": "short","day": "2-digit"}

The typeAttribute section is flexible and allows you to change the format of dates as needed as per below:

Date Type	2-digit	numeric	narrow	short	long
year	✅	✅			
month	✅	✅	✅	✅	✅
day	✅	✅			
hour	✅	✅			
minute	✅	✅			
weekday			✅	✅	✅

See here for examples and more details.

Example 1
With Locale of ‘English (United States)’ and adding the below to the JSON columns:
"typeAttributes": {"year": "2-digit","month": "2-digit","day": "2-digit"}
This will give the date format as below:

2. Close Date
01/17/24
03/13/25
01/31/24

Example 2
With Locale of ‘English (United Kingdom)’ and adding the below to the JSON columns:
"typeAttributes": {"year": "2-digit","month": "2-digit","day": "2-digit"}
This will give the date format as below:

2. Close Date
17/01/24
13/03/25
31/01/24

Currency

This field type uses the type of “currency”. Example JSON:
{"label":"Amount", "fieldName":"record.Amount", "value":"Amount", "type":"currency", "editable":true, "sortable":true, "initialWidth": 200}

Amount
£5,000.00

If you have multiple currencies you can include these by adding the JSON:
"typeAttributes": {"currencyCode": { "fieldName": "record.CurrencyIsoCode"}}
This will give multiple currencies in a column:

ARR
£301.00
US$6,720.00
US$5,760.00

E.g. the above column would be:
*{"label":"Amount", "fieldName":"record.Amount", "value":"Amount", "type":"currency", "editable":true, "sortable":true, "initialWidth":120,
"typeAttributes": {"currencyCode": { "fieldName": "record.CurrencyIsoCode"}},
"cellAttributes": {"alignment": "left" }},*

Number

This field type uses the type of "number". Example JSON:
{"label":"Quantity", "fieldName":"record.TotalOpportunityQuantity", "value":"TotalOpportunityQuantity", "type":"number", "editable":true, "sortable":true, "initialWidth": 200}

Quantity
3

URL

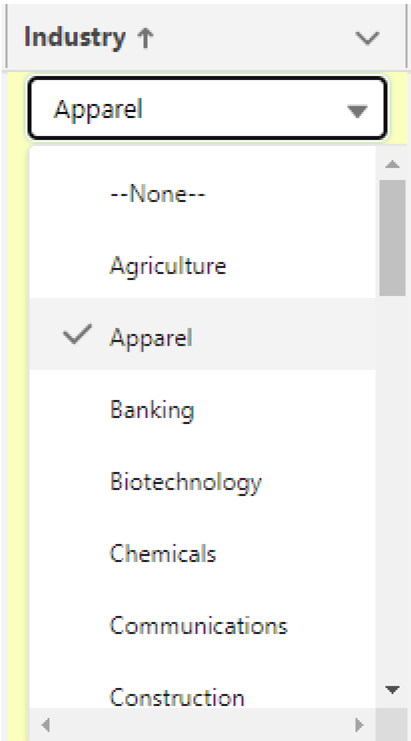
This field type uses the type of "url". Example JSON:
{"label":"Website","fieldName":"record.Website","value":"Website","type":"url","sortable":true,"editable":true, "initialWidth":250}

Website
https://www.asda2.com

Grids by RavenApps contains a number of custom field types to allow you to maximize the power of Grids. These are used automatically when using the Grid Wizard Create and include:

Picklist

This allows picklist functionality to be used for each Grid cell and uses the field type of *ra_picklist*:



Example 1
An example JSON column for this would be for the Type field on Account object :
*{"label":"Type","fieldName":"record.Type","value":"Type","type":"ra_picklist","sortable":true,"editable":true,
"typeAttributes": {"rowId": { "fieldName": "record.Id" },"objectDotField": "Account.Type"}}*

Example 2
For custom objects ensure to use the "__c" notation for the *objectDotField* (not "__r"), e.g. for a custom object of Loan Document this would be *"objectDotField": "Loan_Document__c.Status__c"}*
Full column would beL
*{"label":"Status","fieldName":"record.Status__c","value":"Status__c","type":"ra_picklist","sortable":true,"editable":true,
"typeAttributes": {"rowId": { "fieldName": "record.Id" },"objectDotField": "Loan_Document__c.Status__c"}},*

Picklist allows multi-select in your sandbox () and for Pro licenses in production. For basic licenses in production you can only select a single picklist at a time.

Picklist API name vs label

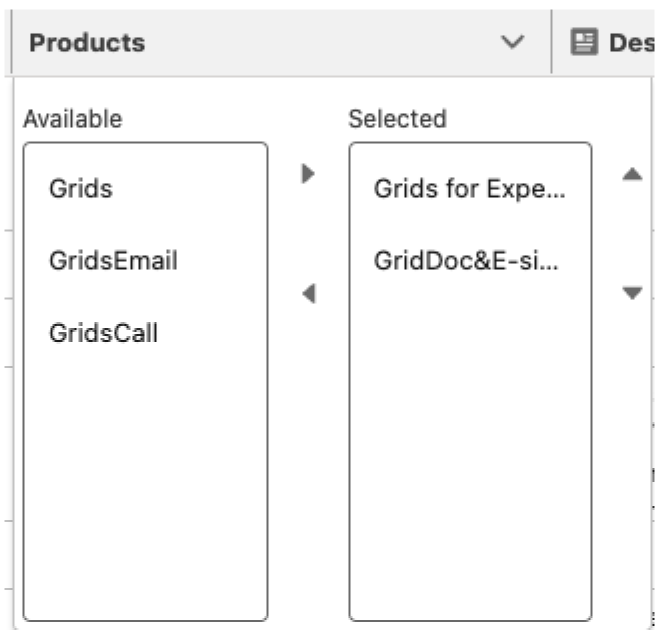
By default grids references the picklist API name, if the label is different to the API name and you wish to reference the label then you can do this by updating your query to use *toLabel()*

For example if you want to query the Account's Type field and return the label then use the query:
SELECT Name, toLabel(Type) FROM Account

There is no need to update the JSON column and this can be left simply as the field name as normal.

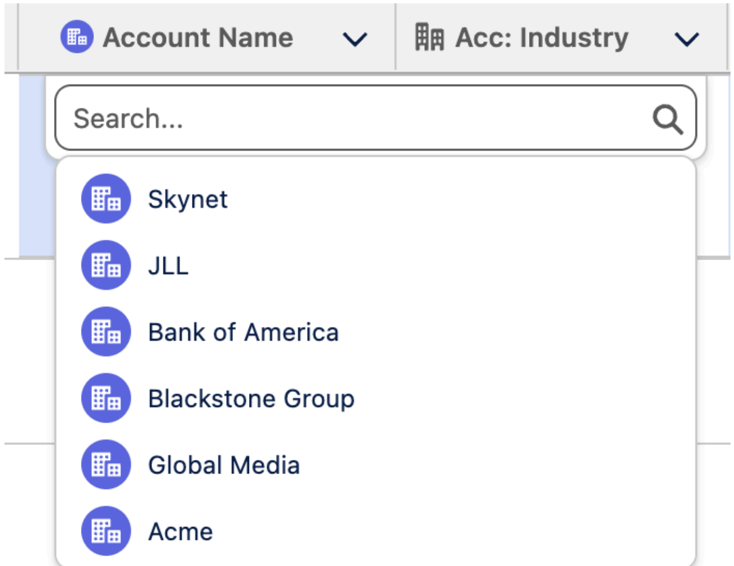
Picklist Multi-Select (DualListBox)

This allows picklist (Multi-Select) functionality to be used for inline edit for column type:



Example:
{"label":"Products", "fieldName":"record.Products__c", "value":"Products__c", "type":"ra_dualListBox", "editable":true, "sortable":true, "iconName":null, "initialWidth":null, "typeAttributes":{"rowId":{"fieldName":"record.Id"}, "objectDotField":"Account.Products__c", "parentRecordId":null}},

RecordPicker (aka Lookup / Master Detail)



Example

{"label":"Account Name", "value":"AccountId","fieldName":"record.AccountId", "editable":true,"iconName":"standard:account","initialWidth":170,"sortable":true,"type":"ra_recordPickerC", "wrapText":true, "typeAttributes":{"valueLabel":{"fieldName":"record.Account.Name"}, "valueLabelField":"Name", "object": "Account", "subTitleField":null, "whereFilter":null, "whereStringVal1":null, "whereIntegerVal1":null, "target":"_self", "icon": "standard:account"}},

Filtered Record Picker Examples (Paid Version)

Hard Coded

{"label":"Account Name", "value":"AccountId","fieldName":"record.AccountId", "editable":true,"iconName":"standard:account","initialWidth":170,"sortable":true,"type":"ra_recordPickerC", "wrapText":true, "typeAttributes":{"valueLabel":{"fieldName":"record.Account.Name"}, "valueLabelField":"Name", "object": "Account", "subTitleField":null, "whereFilter": "Type IN ('Customer - Direct’, 'Customer - Channel')", "whereStringVal1":null, "whereIntegerVal1":null, "target":"_self", "icon": "standard:account"}},

String Variable

{"label":"Contact Name", "value":"ContactId","fieldName":"record.ContactId", "editable":true,"iconName":"standard:account","initialWidth":170,"sortable":true,"type":"ra_recordPickerC", "wrapText":true, "typeAttributes":{"valueLabel":{"fieldName":"record.Contact.Name"}, "valueLabelField":"Name", "object": "Account", "subTitleField":”Account.Name”, "whereFilter": "AccountId =: whereStringVal1", "whereStringVal1":{"fieldName": "record.AccountId"}, "whereIntegerVal1":null, "target":"_self", "icon": "standard:account"}},

Integer Variable

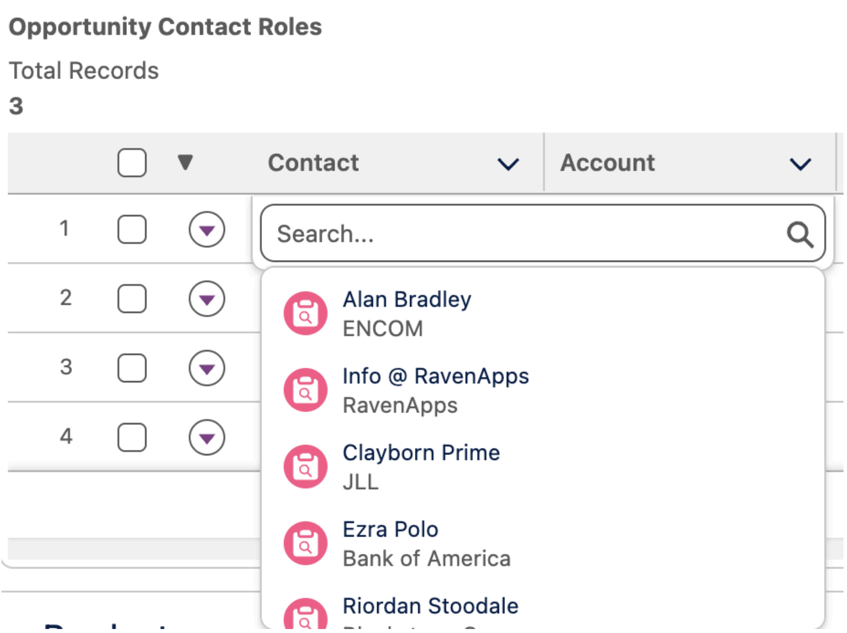
{"label":"Contact Name", "value":"ContactId","fieldName":"record.ContactId", "editable":true,"iconName":"standard:account","initialWidth":170,"sortable":true,"type":"ra_recordPickerC", "wrapText":true, "typeAttributes":{"valueLabel":{"fieldName":"record.Contact.Name"}, "valueLabelField":"Name", "object": "Account", "subTitleField":"Account.Name", "whereFilter": "Account.AnnualRevenue >= :whereIntegerVal1", "whereStringVal1":null, "whereIntegerVal1":100, "target":"_self", "icon": "standard:account"}},

Exclude Current Record (e.g. for Parent Account)

{"label":"Parent", "value":"ParentId","fieldName":"record.ParentId", "editable":true,"iconName":"standard:account","initialWidth":170,"sortable":true,"type":"ra_recordPickerC", "wrapText":true, "typeAttributes":{"valueLabel":{"fieldName":"record.Account.Name"}, "valueLabelField":"Name", "object": "Account", "subTitleField":null, "whereFilter": "Id != :rowId", "whereStringVal1":null, "whereIntegerVal1":null, "target":"_self", "icon": "standard:account"}},

Known Limitation: For inline creation if the variable is referring to the current record being created the filter will not have a value to refer to and hence will not find any values. The work around for this is to either save the record and then update this one field or do not enforce the filter if inline creation is needed. Alternatively can use a flow for record creation.

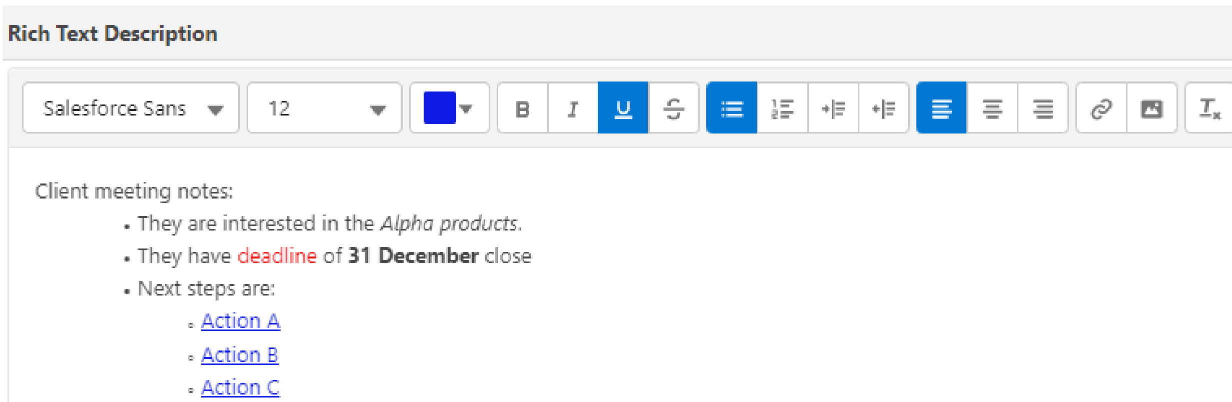
Subtile



{"label":"Contact", "fieldName":"record.ContactId", "value":"ContactId", "type":"ra_recordPickerC", "editable":true, "sortable":true, "iconName":null, "initialWidth":null, "typeAttributes":{"valueLabel":{"fieldName":"record.Contact.Name"}, "valueLabelField":"Name", "object":"Contact", "subTitleField":"Account.Name", "whereFilter":null, "whereStringVal1":null, "whereIntegerVal1":null, "target":"_self", "icon":"standard:record_lookup" } },

Rich Text Area

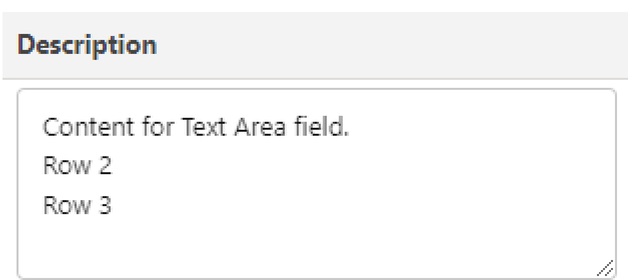
This allows a rich text cells to be used for each Grid cell and uses the field type of *ra_richTextArea*:



An example JSON column for this would be a custom rich text description field:
{"label":"Rich Text Description","fieldName":"record.Rich_Text__c","value":"Rich_Text__c","type":"ra_richTextArea", "sortable":false, "initialWidth": 850, "wrapText":true,"typeAttributes": {"rowId": { "fieldName": "record.Id"}, "field":"Rich_Text__c"}}

Text Area (long text)

This allows a rich text cells to be used for each Grid cell and uses the field type of *ra_textArea*:



An example JSON column for this would be the description field on Account object:
{"label":"Description","fieldName":"record.Description","value":"Description","type":"ra_textArea","sortable":false, "initialWidth": 340, "wrapText":true,"typeAttributes": {"rowId": { "fieldName": "record.Id"}, "field":"Description", "height":"70px", "width":"300px"}}

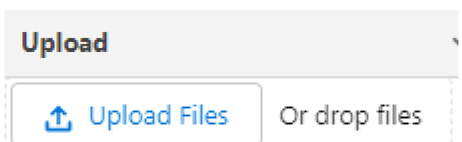
Click To Dial

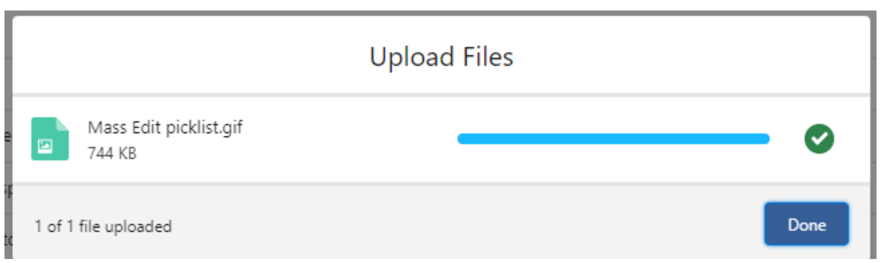
You if wish to activate click to dial from your grid then use the field type of *ra_clickToDial*

An example JSON column for this would be:
{"label":"Contact: Phone", "fieldName":"record.Contact.Phone", "value":"Contact.Phone", "type":"ra_clickToDial2", "editable":true, "sortable":true, "initialWidth": 200, "iconName":"standard:contact"},

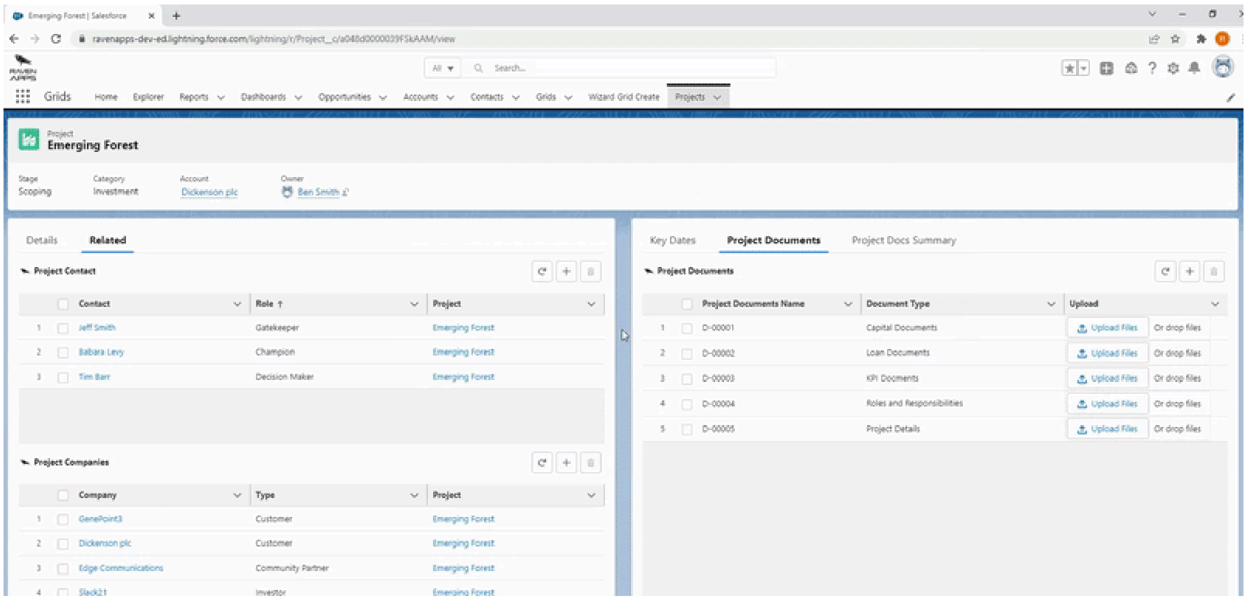
File Upload

This allows a file / files to be drag and dropped onto a cell and these save against the record. The field type is *ra_fileUpload*:





An example JSON column for this would be the description field on Account object:
{"label":"Upload", "type": "ra_fileUpload", "fieldName": "record.Id", "initialWidth": 250, "typeAttributes": {"htmlDisplayAbove": null}}
Example File upload gif:



Troubleshooting & Debugging

If you experience any issues with your grids then please follow these troubleshooting steps:

1. If there is an error message on the grid then review this and see if it provides insights to the cause of the issue.
2. Remember the Grid's:
   a. Query must not contain any carriage returns
   b. JSON is case sensitive - please ensure to check that API names have the correct case.
3. Salesforce does have some objects that in the back end it treats differently to custom objects and most standard objects, these can often be worked around. Some example are:
   a. ContentDocumentLink: Must include a WHERE clause on one of Id, ContentDocumentId, or LinkedEntityId. See more details here
   b. QuoteLineItem: The Search action is not support on this object, to get round this use the Filter action
   c. Events: The following actions are not supported on this object New Record, Quick View, Quick Edit and Inline edit. There are a number of workarounds for this including adding a child object of Events to add new fields that can be edited.

    d. CampaignMember: Search is not supported on thi object.
4. If you are still having issues then please reach out at info@raven-apps.com or book a call here

Supported Field

Grids supports the below field type for read and inline edit capabilities. Please reach out if you need an unsupported area to be available (info@raven-apps.com)

Field Type	Read supported	Inline Edit supported
Lookup Relationship	Y	Y
Master-Detail Relationship	Y	Y
Checkbox	Y	Y
Currency	Y	Y
Date	Y	Y
Date/Time	Y	Y
Time	N	N
Email	Y	Y
Geolocation	Y	Y
Number	Y	Y
Percent	Y	Y
Phone	Y	Y
Picklist	Y	Y
Picklist (Multi-Select)	Y	Y
Text	Y	Y
Text Area	Y	Y
Text Area (Long)	Y	Y
Text Area (Rich)	Y	Y
Text (Encrypted)	N	N
URL	Y	Y
Record Type	Y	Y

Formula	Y	N/A
Autonumber	Y	N/A
Roll up Summary	Y	N/A

Permission Sets

The App comes with the some key standard Permission Sets for its object permissioning:

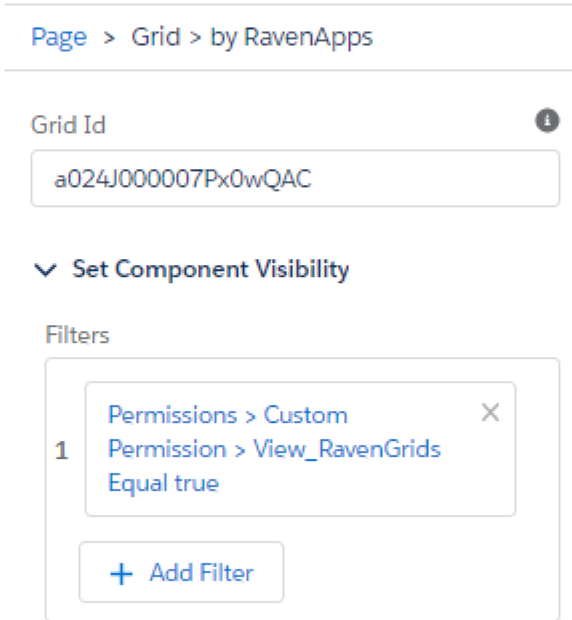
<u>Objects:</u> Grids, Grid Views, User Grid Views and Usage Insights

It consists of the 3 permissions sets:

Permission Set	Aimed at
RavenGrids Admin User	Full system admin privileges to create, edit and delete grids
RavenGrids Create and Edit Users	End users who will be using grids and be able to use all functionality of grids.
RavenGrids View User	End users who will be using grids but not creating or editing any themselves. They will have the ability to use all features on the grids (inline edit, filters, sorts, search etc…) except for Grid Views where create and / or edit functionality is needed

Custom Permission: RavenApps - Grids Viewer

All the permission sets contain a custom permission called 'RavenApps - Grids Viewer' which can be used to set Grid components dynamic visibility so user without Grids access don't see components they don't have access to. This can be set up on the each Grids component Lightning page as below:



FAQs

Sandbox to Production Data Migration

Dataload

1. Dataload sandbox Grid records - could use [workbench](), [dataloader]()
   - recommend prior setting the Grid records UID__c (Grid Id) field to a friendly name e.g. AccountGrid and also updating Lightning Pages which reference it.
2. Assign users a 'Raven...' permission
3. In **Setup** -> **Installed Packages** -> **Grids** (RavenApps) -> **Manage Licenses** - assign to users

Manual Copy & Post

Force Newline in Freeze Column Header to Sync with Unfreeze Column

Add ‘\n’ in the name for example: Column Label = Account \n Name

Feature Details

Freeze Columns

When many columns it can be useful to freeze the first n columns to give context across the whole grid. To enable freeze column simple follow the below steps:

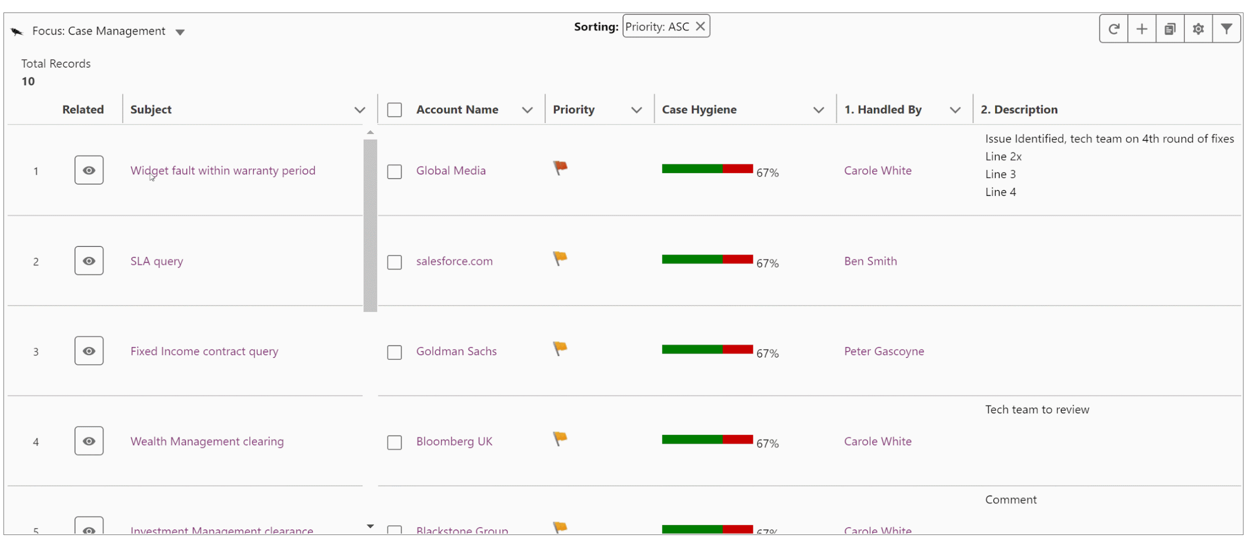
- Ensure all columns have an initialWidth set.

- Populate the ‘Number of Frozen Columns’ field - This will define how many columns are frozen.

- Populate the ‘Row Height’ field - This generally only needs to be populated in the below scenarios:

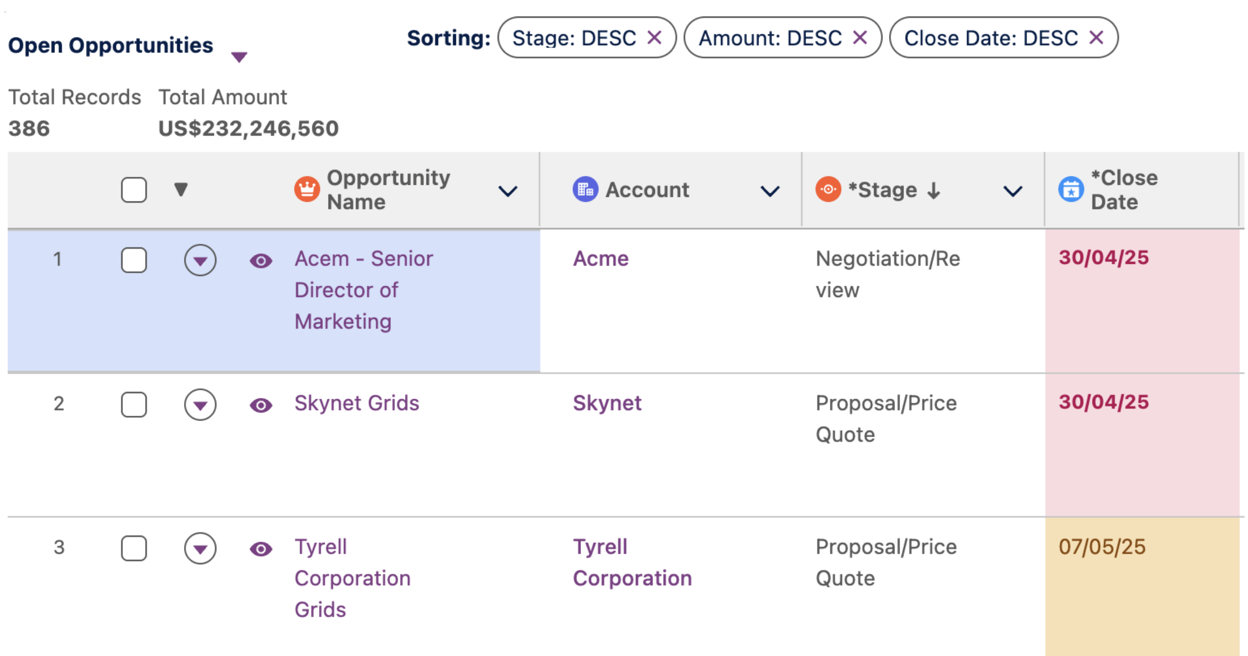
    - Text Area field type: populate the ‘Row Height’ with the same value as the column height
    - Rich Text Area: use a ‘Row Height’
    - If not using Text Area or Rich Text Area field types you can populate ‘Row Height’ to increase the row height if useful
    - If any other issues with the rows not aligning between grids try entering a height of 30

In the below Grid the Number of Frozen Columns is set to 2 and the Row Height is set to 90 to match the Description fields Text Area height:



When working with Grids on mobile with freeze column you may wish to use your mobile on landscape mode (see here)

Conditional Formatting



1. Create a formula field to logically give the conditional style class(s) outcomes desired

   Theme Classes (includes background and text color)

   .slds-theme_success - Sets the background color to green
   .slds-theme_warning - Sets the background color to yellow
   .slds-theme_error - Sets the background color to red

   Text Styles / Sizes classes

   Example
   Based on Opportunity Close Date this field could be called ‘Close Date CF’-‘CloseDateCF__c’

   IF(AND(CloseDate < TODAY(), IsClosed = FALSE), "slds-theme_error slds-text-title_bold",
   IF(AND(CloseDate < (TODAY() + 14), IsClosed = FALSE), "slds-theme_warning",""))

2. Add the below JSON referring the formula field to any column where we want the conditional formatting to be

   {"label":"*Close Date", "fieldName":"record.CloseDate", "value":"CloseDate", "type":"date-local", "editable":true, "sortable":true, "initialWidth": 120, "iconName": "standard:key_dates", "wrapText":true,
   "typeAttributes": {"year": "2-digit","month": "2-digit","day": "2-digit"},
   "cellAttributes": {"class": { "fieldName": "record.'CloseDateCF__c" }}
   },

3. Add the new formula field to the Grid's Query field

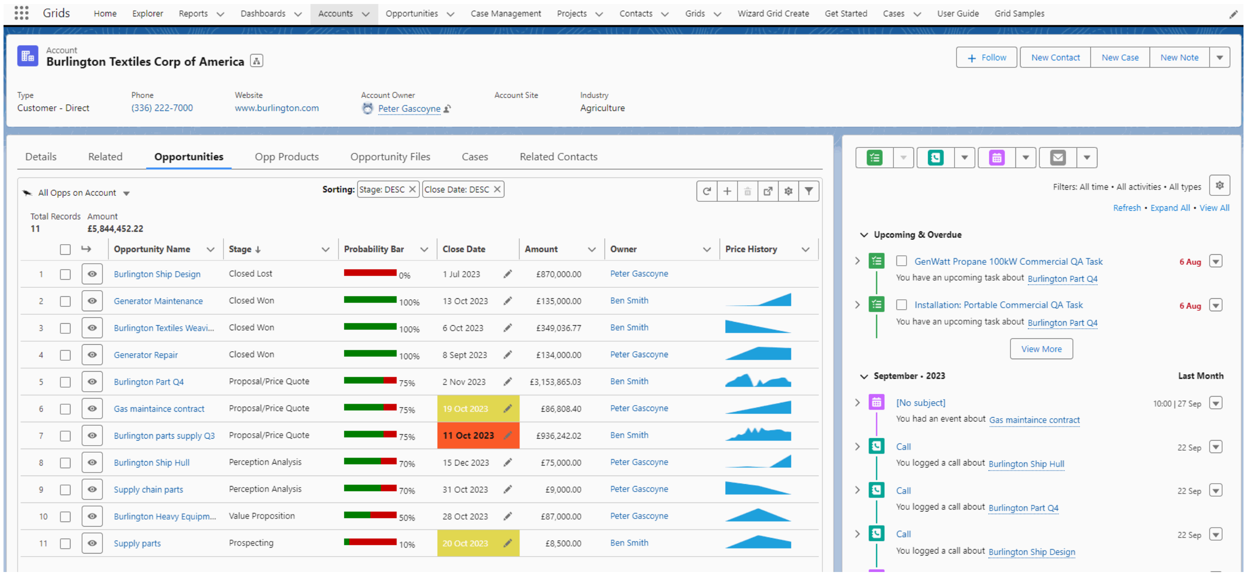
   SELECT Name, Owner.Name, AccountId, Account.Name, Amount, ExpectedRevenue, CloseDate, Probability, StageName, NextStep, LeadSource, Type, TotalOpportunityQuantity, toLabel(ForecastCategoryName), Account.Industry, Description, CreatedDate, CreatedBy.Name, Opportunity_Hygiene__c, CloseDateCF__c FROM Opportunity

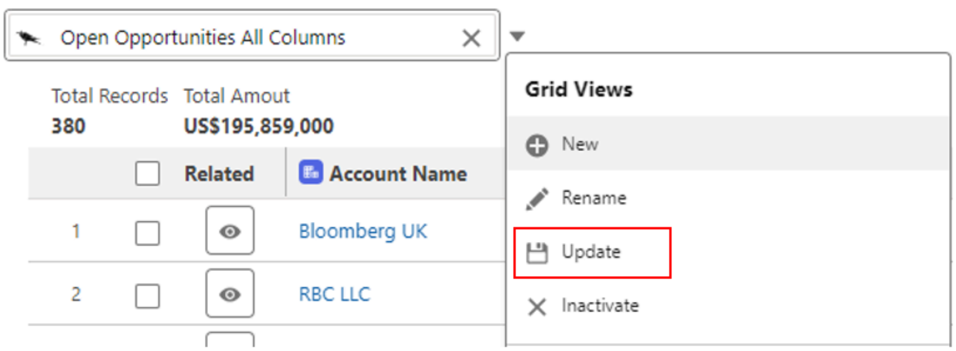
Grid Views

Grid Views allow the end user to be able to update the columns selected, filters and sorting of a grid and save view so they and others can use this view in the future.

Add the Action 'Grid Views' to any Grid.

If you want your end users to be able to create and or Grid View you will need to ensure they have permission to do this. The [RavenGrids Create and Edit Users](#) permission set can be used or you can configure your own permissions.



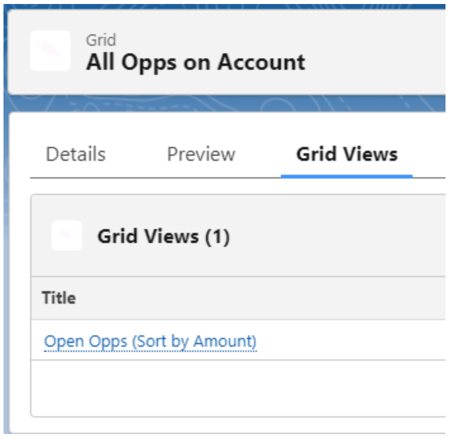


*Please note you can not update a Primary Grid from this drop down, only the Grid View.

Grid View made inactive will no longer be seen in a grid views list.

In the backend the Grid View is a new Grid_c record that has the ‘Primary Grid’ field (‘*RavenApps__Primary_Grid__c*’) populated with the original Grid. There can only be 2 levels, so if a Grid View is created off an existing Grid View the Primary Grid will be the first Grid View's Primary Grid.

On the Primary Grid you can see all associated Grid views and these will all be visible on the Grid:

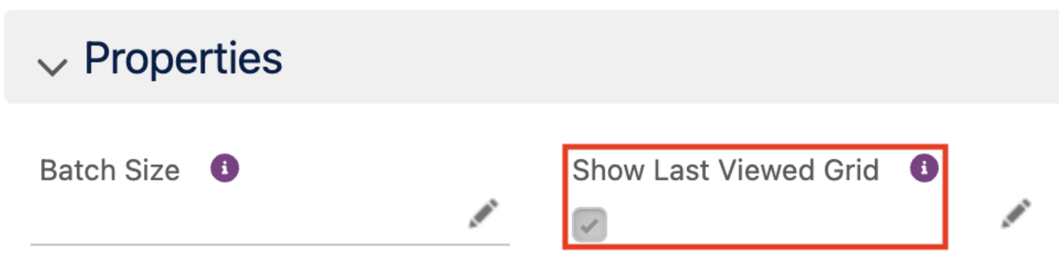


At present you can save Sorts and Filters, more features coming soon and can be fast tracked upon request (info@raven-apps.com).

Views - Show Last Viewed Grid

Ability to default the Grid which displays for users based on what they last viewed:

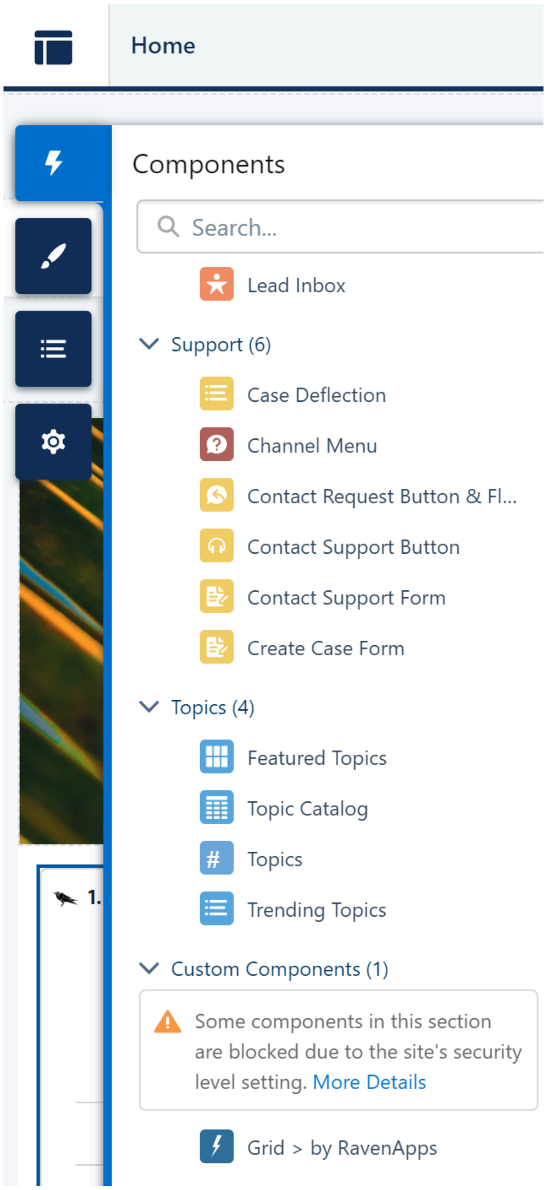
Show Last Viewed Grid



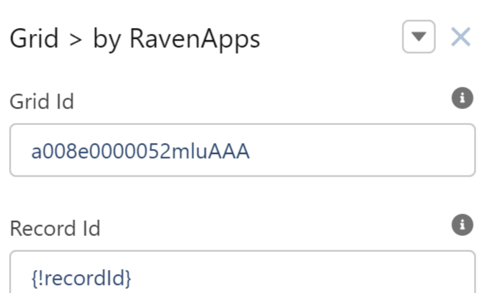
Experience Cloud

From version 4.127
Grids Experience Cloud license required

Adding Grids to Experience Cloud pages
In the Experience Cloud builder select the Components icon and drag the ‘Grid > by RavenApp’ component onto the page



Click on the ‘Grid > by RavenApp’ component and add the Grid id. You can also optionally add the Record Id if the Grid is a Related List Grid that is dependent on the record page you are on, simply enter {!recordId} into the box.



Other considerations

You will need to ensure Experience Cloud users have the relevant permissions. this will include:

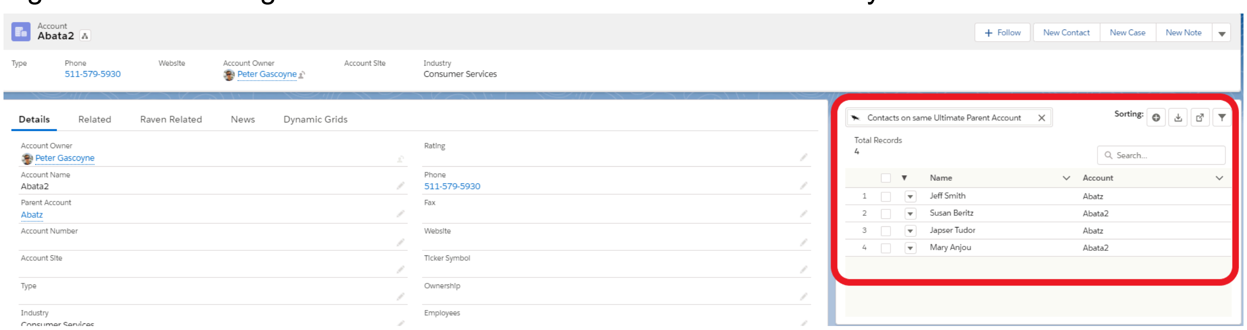
- A Grids license - see point 2 here
- Access to the Grids records, for example you can do this through:
    - Sharing Setting -> Grids -> set Default External Access from Private to Read Only
- Access to the relevant Salesforce records as required in all Experience Cloud sites.

Read More

Where can you place grids?

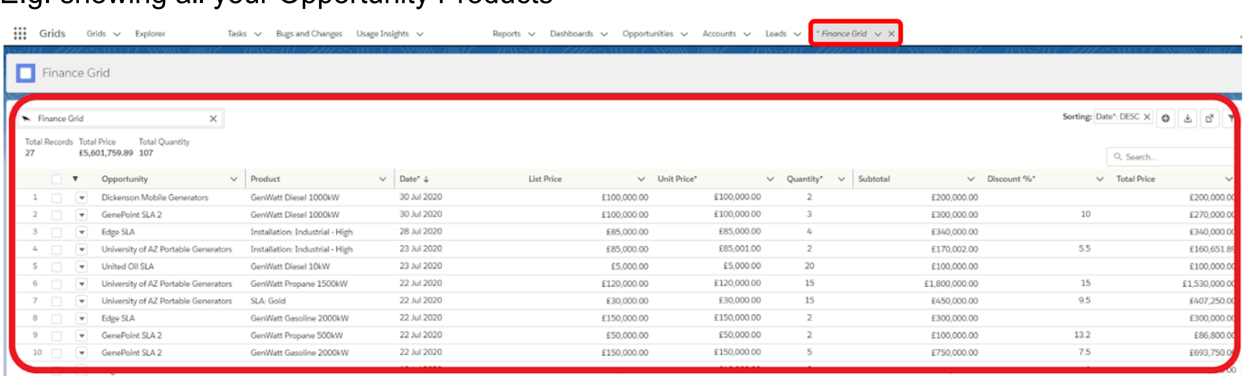
1. Record Pages

E.g. such as showing all Contacts on all Accounts within a hierarchy:



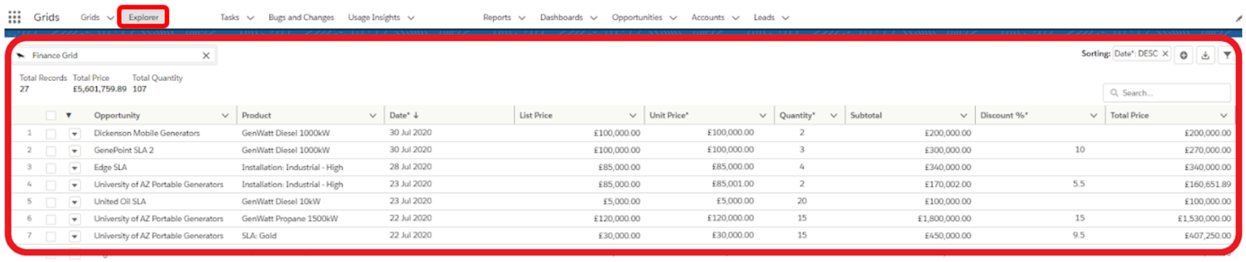
2. Lightning App Pages

E.g. showing all your Opportunity Products



3. RavenApps Explorer

This is a tab that allows you to quickly switch between all grid that have ‘Visible on Explorer’ checked true.

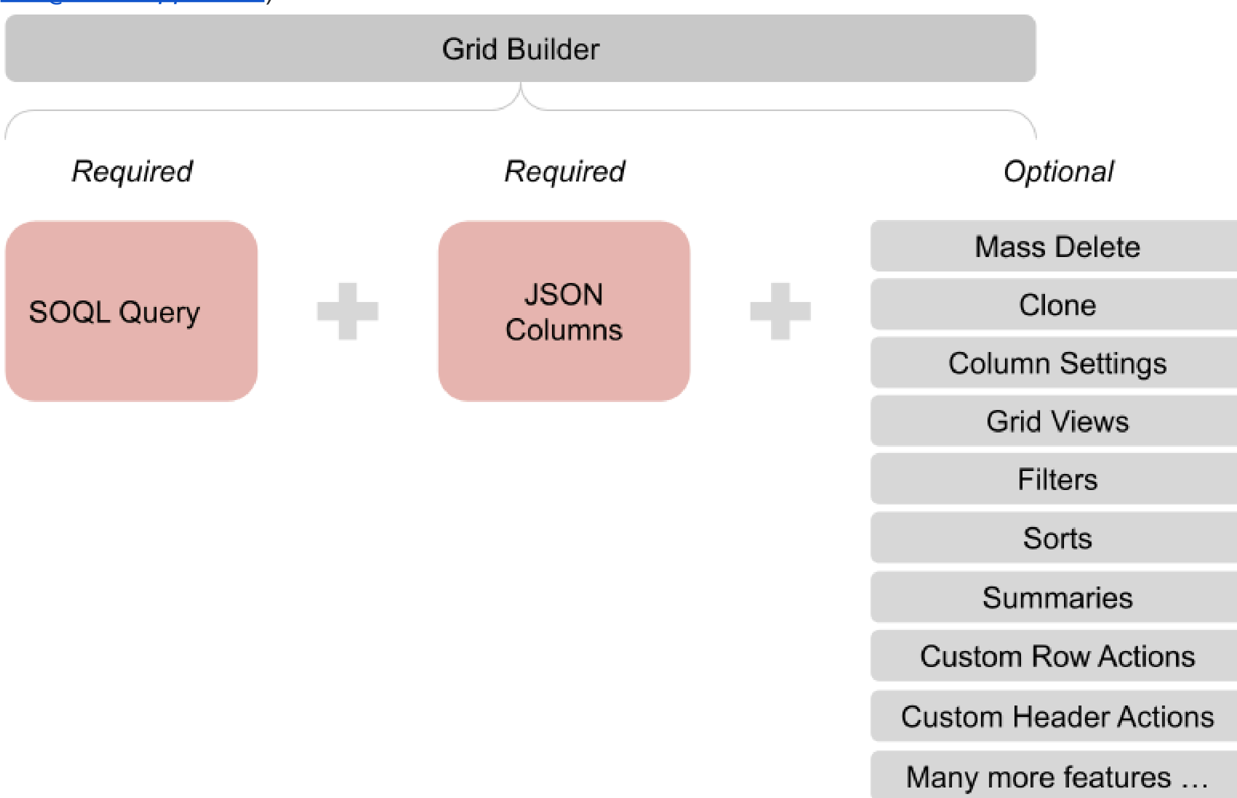


	Opportunity	Product	Date*	List Price	Unit Price*	Quantity*	Subtotal	Discount %*	Total Price
1	Dickenson Mobile Generators	GenWatt Diesel 1000kW	30 Jul 2020	£100,000.00	£100,000.00	2	£200,000.00		£200,000.00
2	GenePoint SLA 2	GenWatt Diesel 1000kW	30 Jul 2020	£100,000.00	£100,000.00	3	£300,000.00	10	£270,000.00
3	Edge SLA	Installation: Industrial - High	28 Jul 2020	£85,000.00	£85,000.00	4	£340,000.00		£340,000.00
4	University of AZ Portable Generators	Installation: Industrial - High	23 Jul 2020	£85,000.00	£85,001.00	2	£170,002.00	5.5	£160,651.89
5	United Oil SLA	GenWatt Diesel 10kW	23 Jul 2020	£5,000.00	£5,000.00	20	£100,000.00		£100,000.00
6	University of AZ Portable Generators	GenWatt Propane 1500kW	22 Jul 2020	£120,000.00	£120,000.00	15	£1,800,000.00	15	£1,530,000.00
7	University of AZ Portable Generators	SLA: Gold	22 Jul 2020	£30,000.00	£30,000.00	15	£450,000.00	9.5	£407,250.00

The application has been built to align with the latest Salesforce LWC (Lightning Web Component) framework so as to align and benefit from future developments from Salesforce.

Grid Builder

Each Grid is represented by a Salesforce record which must have a SOQL Query and at least 1 Column using JSON (a text-based format for representing structured data) and there are many additional optional features that can be added in. There is a Grid wizard builder that allows you to create the SOQL Query and JSON columns using clicks; for some of the more advanced features entering JSON directly is required and this user guide takes you through the structure of these with some examples (we also respond very quickly to any questions at *info@raven-apps.com*).



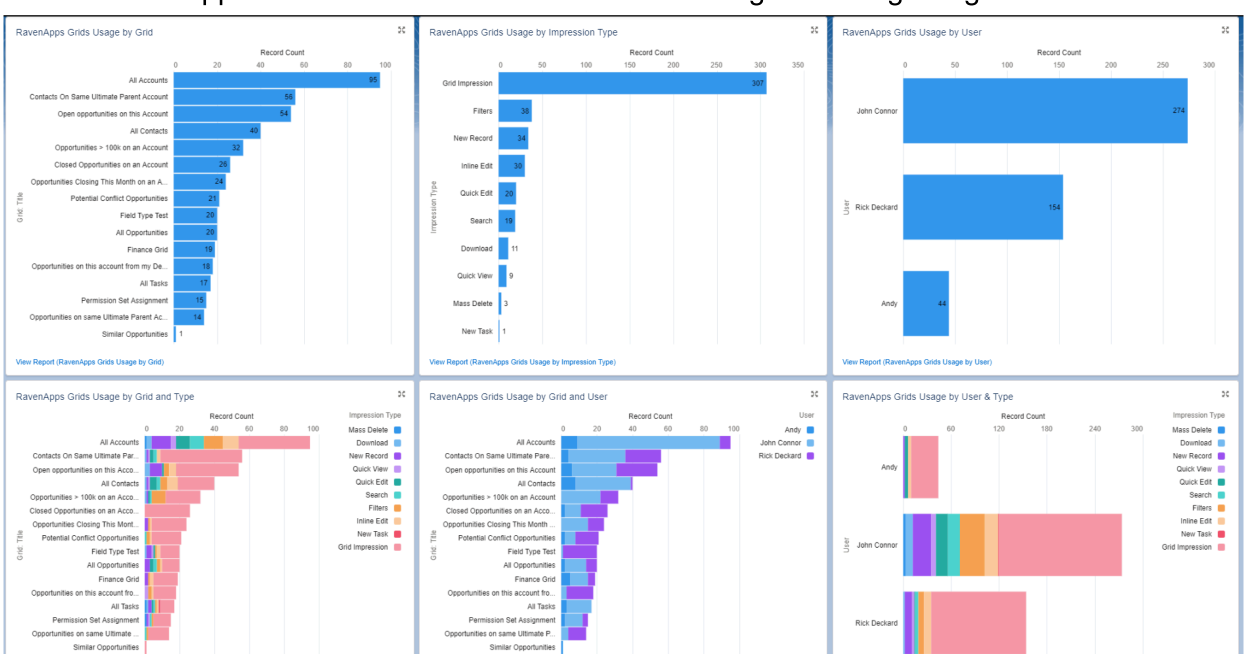
Usage Insights

To measure the usage of each grid everytime a grid is interacted with a Usage Insight record is created detailing the grid, the user, the impression type and the date. To avoid unnecessary build up of data a maximum of 1 record is created per user per grid per impression per day. E.g. If a user views a grid 5 times in a day this will show up as 1 Usage Insight record of Impression Type ‘Grid Impression’.

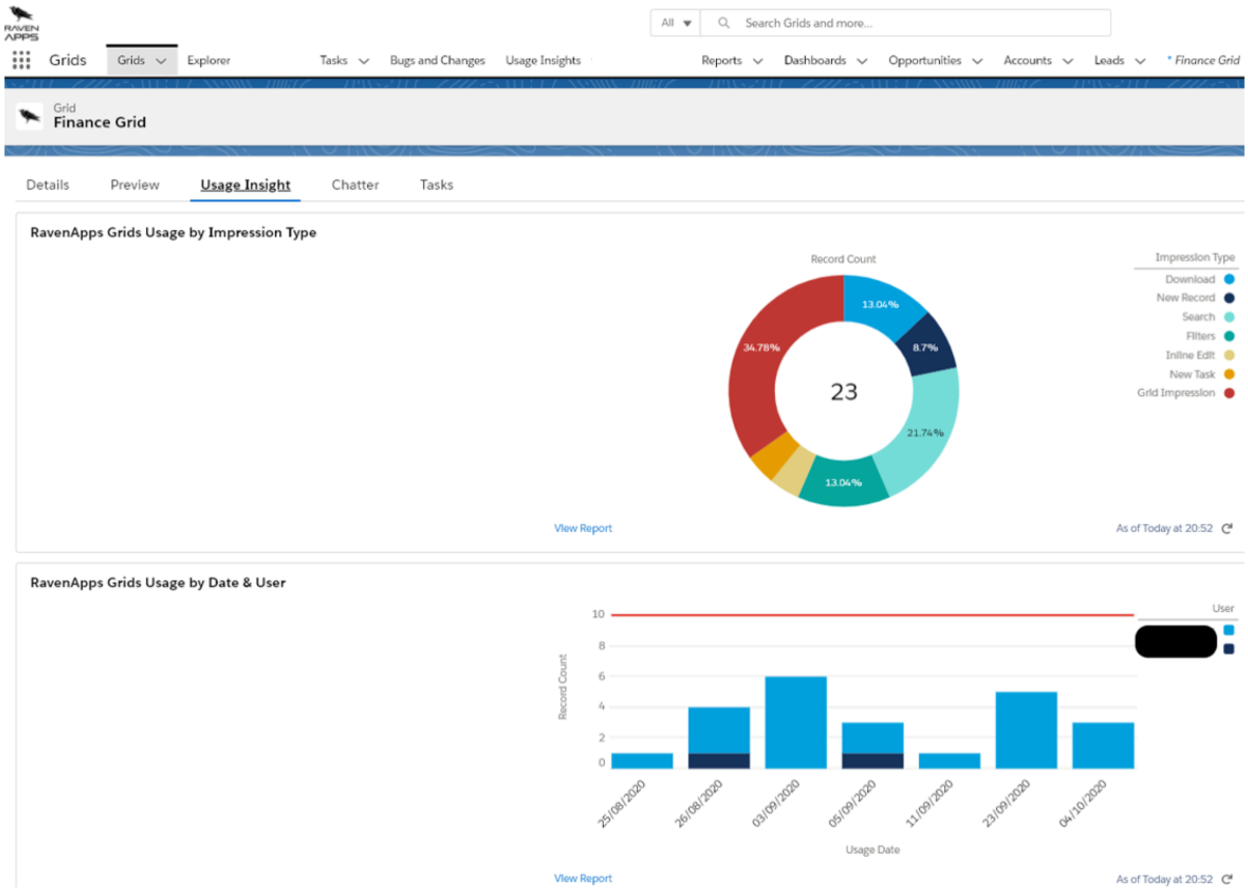
Types of Impression are mainly aligned to the actions available on the Grids with the addition of ‘Inline Edit’ and ‘Grid Impression’:

- Delete Record
- Mass Delete
- Download
- New Record
- Quick View
- Quick Edit
- Search
- Filters
- Inline Edit
- New Task
- Grid Impression

These records are then viewable in the reports folder ‘RavenApps Reports’ and Dashboard Folder ‘RavenApps Dashboards’ that come with the Managed Package. E.g.



On each Grids record you can also view the Usage Insights tab to see who is using the Grid for what purpose and when:



Grid Actions

Defines which actions are available on the Grid for the user. Standard Salesforce access permissions are followed so users can only CRED (create, read, edit and delete) as per their allowed Salesforce access.

Action	Description
Delete Record	Allows delete of a record through the drop down selection
Mass Delete	Allows multiple records to be selected and deleted using the delete icon at the top right of the page
Download	Allows the grid data to be downloaded to csv using the download icon at the top right of the grid

New Record	Allows creation of a new record for the main object type at the top right of the grid
Quick View	Allows individual records to be opened to view in a modal from the record drop down
Quick Edit	Allows individual records to be opened to edit in a modal from the record drop down
Search	Includes the search box at the top right of the grid. Remember you can use the wild cards * and ? to help your search (more detail here). Search functionality covers all indexed fields (see Salesforce standard indexed field here). For custom fields all of the below field types are automatically indexed and hence searchable: ● Name ● All custom auto-number fields and custom fields that are set as an external ID (no need to enter leading zeros) ● All custom fields of type email and phone ● All custom fields of type text, text area, long text area, and rich text area Records will be returned if the match is found in the query, even if the column is not on the page.
Filters	Defines if the filter button is available to users. The filters functionality below will function regardless of if this action is included for the user. Filters are generally recommended over search as they cover all matches while Salesforce Search is not always clear what fields it is searching.
Full Screen	This allows users to expand a grid to be full screen by clicking on an icon at the top right of the grid. This will show on grids created through the custom component 'Grids > by RavenApps' but not on Explorer Tab as these are already full screen.
Go To Record	Allows user to click the drop down arrow and click to go to the record
Sorting	Allows the pre-set sorting to be viewed at the top right of the page and for the user to remove these individually.

Summaries	Allows summaries to be displayed at the top of the page (see Summaries section below)
New Record (Row)	Allows new records to be created inline in the grid, all required fields must be on the grid for this to work. It can be used in conjunction with New Record Row Defaults to auto populate lookup fields or other fields
Column Settings	Allows end users to add, remove and rearrange columns, see more details here
Clone	Allows a line(s) to be cloned on a grid, see more details here
Grid Views	Allow end users to create their own grid views, see more details here
Editable Parent Fields	Allow parent fields to be edited in the grid, see more details here

These actions appear at the top right of the grid when selected. For Example:

Some actions will not work for some objects due to Salesforce limitations. RavenApps have selectively enhanced some of these areas and welcome feedback on other areas that are required (info@raven-apps.com).

Query

The SOQL is automatically created by the Grid Wizard but you can subsequently update to add in extra fields such as parent fields.. All fields referenced in Columns, Sorts, Filter and Summaries must be included in the SOQL query. The SOQL query can be tested in Workbench or the developer console to check the query is working correctly.

Columns

Columns are defined in the Grid using JSON. The minimum information for each record should be Label and FieldName. E.g. for each field
{"label":"Name","fieldName":"record.Name","value":"Name}
“Value” is needed in addition to “fieldName”.

Other details can be included such as Type, Sortable, Editable, initialWidth, Alignment and much more. (see Appendix 1 for more details). For example:
{"label":"Quantity*","fieldName":"record.Quantity","value":"Quantity","type":"number","sortable":true,"editable":true,"initialWidth":120,"cellAttributes": { "alignment": "center" }},
{"label":"Subtotal","fieldName":"record.Subtotal","value":"Subtotal","type":"currency","sortable":true,"editable":false}
In the above example 2 fields, Quantity and Subtotal, have been defined and are separated by a comma after the curly brackets have closed.

Initial Width

By default columns have no initial width so will dynamically expand to the page size. To set an initial width on a column simply add the below into the columns JSON:
"initialWidth": 150

Example
{"label":"Quantity*","fieldName":"record.Quantity","value":"Quantity","type":"number","sortable":true,"editable":true,"initialWidth":150}

Example 2
For ra_textArea field you will need to set both the initialWidth and the width attribute with the later being slightly less than the former:
{"label":"Description", "fieldName":"record.Description", "value":"Description", "type":"ra_textArea", "sortable":true,"wrapText":"true", "initialWidth":522, "typeAttributes": {"rowId": { "fieldName": "record.Id" }, "field":"Description", "height":"50px", "width":"490px" }}

Read Only / Non editable Columns

To make a column non editable simply use the JSON "*editable":false*
Example
{"label":"Quantity","fieldName":"record.Quantity","value":"Quantity","type":"number","editable":false}*

For any custom field type (picklist, lookup, name, long text, rich text, etc..) columns if you wish to make these not editable in the grid then you will need to add the JSON section *"editable":false* within the *typeAttributes* section of the JSON.
Example 2
{"label":"Stage", "fieldName":"record.StageName", "value":"StageName", "type":"ra_picklist","typeAttributes": {"rowId": { "fieldName": "record.Id" },"editable":false,"objectDotField": "Opportunity.StageName"}}
Example 3
{"label":"Name","fieldName":"record.Name","value":"Name","type":"ra_name", "sortable":true, "initialWidth": 200, "typeAttributes": {"rowId": { "fieldName": "record.Id" },"editable":false}}

Sorts

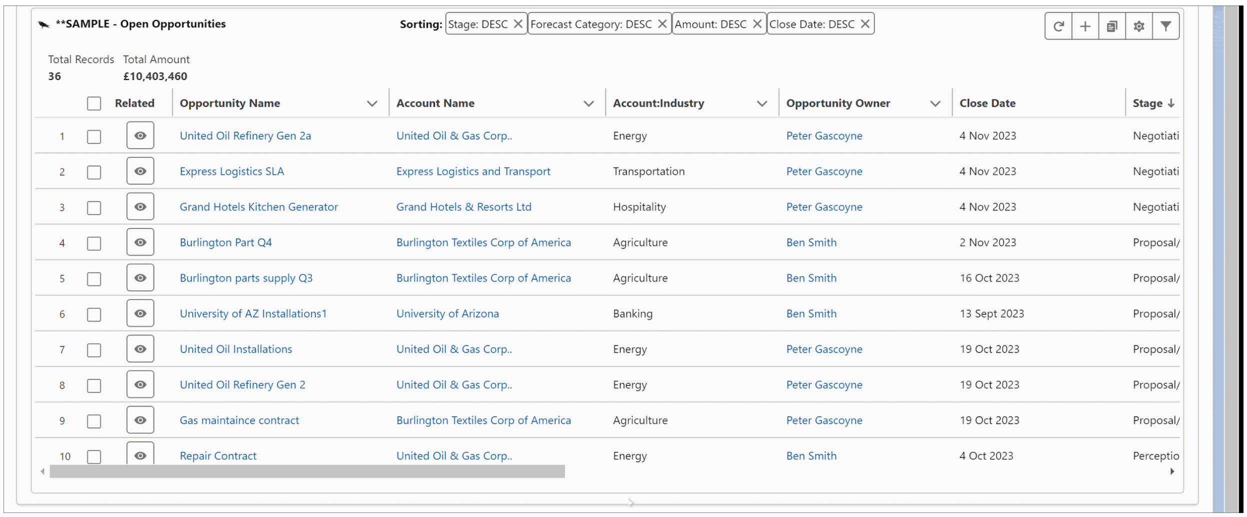
Define the default sorting of the records presented using JSON. It allows multiple sorting, below is example for sorting Case based on Rating descending then Name ascending:

{"order":1, "label":"Rating: ASC", "field":"Rating", "direction": "*asc*"},

{"order":2, "label":"Name: ASC", "field":"Name", "direction": "asc"}

You can define at the column level if you want a column to be sortable or not. By default it will be sortable but you can restricted this by using adding the JSON *"sortable":false*
E.g.
{"label":"Subtotal","fieldName":"record.Subtotal","value":"Subtotal","type":"currency","sortable":false}

Example
The below grid has 4 pre-sorts and then the end users can remove these or add new sorts:


**SAMPLE - Open Opportunities   Sorting: Stage: DESC Forecast Category: DESC Amount: DESC Close Date: DESC

Total Records 36   Total Amount £10,403,460

	Related	Opportunity Name	Account Name	Account:Industry	Opportunity Owner	Close Date	Stage
1		United Oil Refinery Gen 2a	United Oil & Gas Corp..	Energy	Peter Gascoyne	4 Nov 2023	Negotiati
2		Express Logistics SLA	Express Logistics and Transport	Transportation	Peter Gascoyne	4 Nov 2023	Negotiati
3		Grand Hotels Kitchen Generator	Grand Hotels & Resorts Ltd	Hospitality	Peter Gascoyne	4 Nov 2023	Negotiati
4		Burlington Part Q4	Burlington Textiles Corp of America	Agriculture	Ben Smith	2 Nov 2023	Proposal/
5		Burlington parts supply Q3	Burlington Textiles Corp of America	Agriculture	Ben Smith	16 Oct 2023	Proposal/
6		University of AZ Installations1	University of Arizona	Banking	Ben Smith	13 Sept 2023	Proposal/
7		United Oil Installations	United Oil & Gas Corp..	Energy	Peter Gascoyne	19 Oct 2023	Proposal/
8		United Oil Refinery Gen 2	United Oil & Gas Corp..	Energy	Peter Gascoyne	19 Oct 2023	Proposal/
9		Gas maintaince contract	Burlington Textiles Corp of America	Agriculture	Peter Gascoyne	19 Oct 2023	Proposal/
10		Repair Contract	United Oil & Gas Corp..	Energy	Ben Smith	4 Oct 2023	Perceptio

The JSON for the 4 Sorts above is:
{"order":1, "label":"Stage: DESC", "field":"StageName", "direction": "desc"},
{"order":2, "label":"Forecast Category: DESC", "field":"ForecastCategoryName", "direction": "desc"},
{"order":3, "label":"Amount: DESC", "field":"Amount", "direction": "desc"},
{"order":4, "label":"Close Date: DESC", "field":"CloseDate", "direction": "desc"}

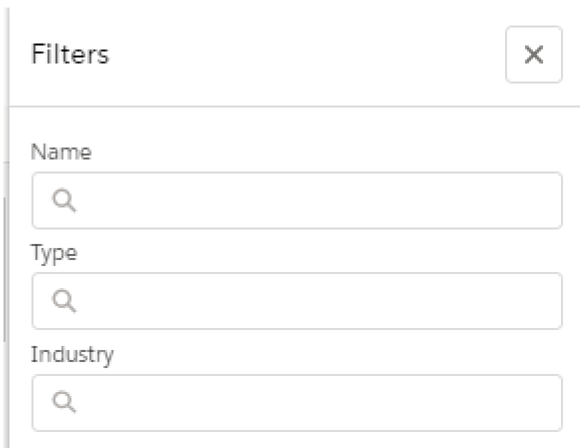
Filters

Populate this field with JSON to filter the data before it is presented in the Grid (this is on top of any filters applied in the Query.
To populate the filter include:

1. Label
2. Field
3. Value
4. Function, options are:
    a. =
    b. !=
    c. >
    d. <
    e. >=
    f. <=
    g. *Contains*

5. Type: options are:
    a. *Text*
        i. Supports comma separated filters. E.g. for the Account filter using “Apple, Google” will return both values (or any other values containing these if using contain function)
    b. *picklist*
    c. *datetime*
    d. *date*
    e. *checkbox*
    f. *number*
    g. *currency*

Blank filters can be used by setting the value to “” or null so that the filter is then available on the Grid. E.g.for the these 3 filters
{"order":1,"label":"Name","field":"Name","value":"","function":"contains","type":"text"},
{"order":2, "label":"Type","field":"Type","value":null,"function":"contains","type":"text"},
{"order":3, "label":"Industry","field":"Industry","value":null,"function":"contains","type":"text"}
The Grid will display the below when the filter button is clicked on:



Filter logic is automatically set to AND across so that all filters included will all need to be met for records to be included in the grid. If an OR statement is needed then it is suggested to use a formula field that includes OR logic and base the filter on this field. E.g. to create a filter on records created in the last 7 days with priority equal to ‘High’ use this formula field equals
IF(OR(DATEVALUE(CreatedDate) > TODAY() - 7,ISPICKVAL(Priority,"High")),"Yes","No")

Filtering using Variable

Filters can also reference [Variables] which is useful when wanting the Grid to only display records with some reference to the current record. E.g. On the Opportunity page you could put a Grid showing all other opportunities that are selling the same product
{"label":"Product", "field":"Product_Name__c", "value":"sv1", "valueType":"variable", "function":"=","type":"text"}

Here sv1 would be defined as {"type":"record", "field":"Product_Name__c", "variable":"sv1"}

Records will be returned if the match is found in the query, even if the column is not on the page.

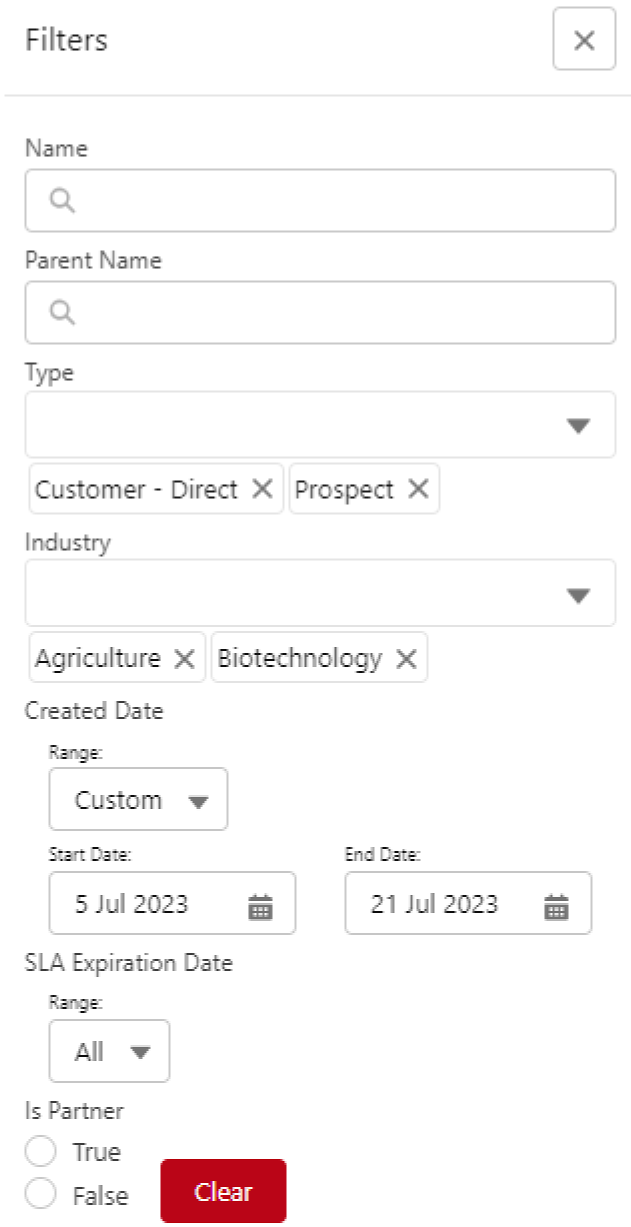
Filtering by Field Types

Filters support the below field types:

- text:
    - E.g. {"order":1,"label":"Name","field":"Name","value":"","function":"contains","type":"text"}
- date
    - E.g. {"order":1, "label":"Created Date","field":"CreatedDate","value":"","function":"=","type":"datetime","valueType":"literal"}
- picklist
    - Simply use type of 'picklist' and define the picklist by making the objectDotField reference the picklist you need, e.g.:
        - {"order":1,"label":"Industry","field":"Industry","value":null, "function":"contains","type":"picklist","objectDotField": "Account.Industry"}
    - When using custom objects the objectDotField should use "__c." notification (not "__r."), e.g. for the custom object Bid:
        - {"order":1,"label":"Offer Stage","field":"Offer_Stage__c","value":null, "function":"=","type":"picklist","objectDotField": "Bid__c.Offer_Stage__c"},
    - You can also pre set picklist values using comma separated values as below for values Pipeline,Omitted,Best Case,Commit,Closed, e.g.:
        - {"order":2, "label":"Forecast Category","field":"ForecastCategoryName","value":"Pipeline,Omitted,Best Case,Commit,Closed","function":"contains","type":"picklist","objectDotField": "Opportunity.ForecastCategoryName"}
- boolean
    - Simply mark the type as checkbox:
    - E.g. {"order":2, "label":"IsPartner","field":"IsPartner","value":null, "function":"=","type":"checkbox"}
- number
    - {"order":1, "label":"Employees (> than)","field":"NumberOfEmployees", "value":0,"function":">","type":"number"}
- currency
    - {"order":2, "label":"ARR (>=)","field":"Amount","value":null,"function":">=", "type":"currency"}
- Lookup
    - For lookup filters simply use the text filter and reference a field on the parent record. You can also filter by grandparents etc..
    - An example of filtering by parent account when on an opportunity would be:

- *{"order":1, "label":"Account", "field":"Account.Name", "value":"", "Function":"contains", "type":"text"},*

Filter then appear for example as below:



Pre-Filtering by Relative Date Values

When filtering by date you may wish to filter relative to today's date, e.g. filter by last month. You can do this using Salesforce standard relative date filters. The [full list is here](#).
Examples would be LAST_MONTH, LAST_90_DAYS, NEXT_N_MONTHS:n etc…

e.g.:
{"order":1, "label":"Created Date","field":"CreatedDate","value":"THIS_QUARTER", "function":"=","type":"datetime","valueType":"literal"}

Pre-Filtering by Absolute Date Values

You can also filter by absolute values if needed. An example would be:
{"order":1, "label":"Created Date","field":"CreatedDate","value":"", "function":"=","type":"date", "startDateValue":"2022-06-15", "endDateValue":"2022-12-31"}

Summaries

Include which fields should be summarized in the Grid using JSON. For each record this must include the Label, Field, Type and formatStyle. E.,g.
{"label": "Total Price", "field":"TotalPrice", "type": "SUM", "formatStyle": "currency"},
{"label": "Total Quantity", "field":"Quantity", "type": "SUM", "formatStyle": "decimal"}

You can use the following operators for summaries:

- SUM
- COUNT
- COUNT_DISTINCT
- MAX
- MIN
- AVG

If you have a grid with >50,000 rows and you include a summary you will get a 'Too many query rows: 50001" error. If you get this error you can either update filters to reduce the query size, remove summaries all together or reach out to our team to see if there is another solution (info@raven-apps.com)

Variables

Variables are populated using JSON when the filter or the query needs to reference the current record (or related records). E.g. when trying to show all Opportunities that have the same Industry as the current Opportunity's Account's Industry then use the variable definition below.
{"type":"record", "field":"Account.Industry", "variable":"sv1"}
Use the variable names 'sv1' or 'sv2'.

You will find a number of Example Grids in the installed Managed Package and there are also some examples in the appendix of this document.

Add Parent Fields

If you wish to add parent / grandparent fields into a grid then just use the [standard Salesforce syntax](#) for parent fields.

Example 1

For a Contact Grid pull in the Account's Industry and Customer Priority fields using the below query:

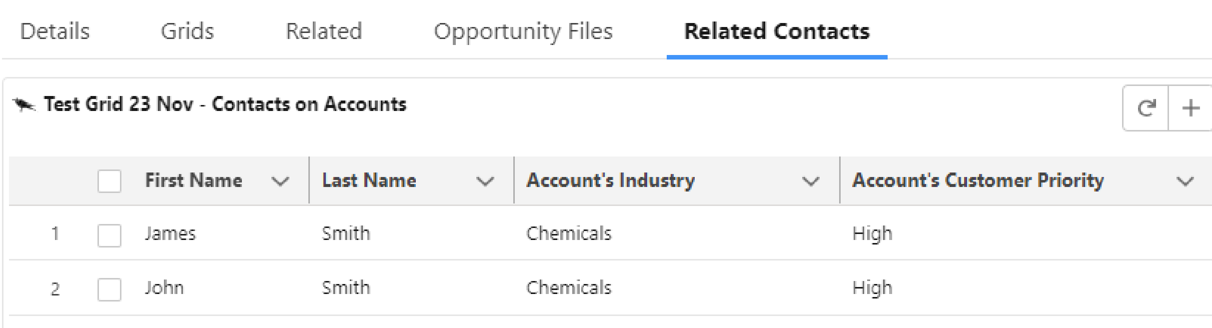
SELECT FirstName, LastName, Account.Industry, Account.CustomerPriority__c FROM Contact WHERE AccountId =: sv1

Then the below columns:

{"label":"Acc: Industry", "fieldName":"record.Account.Industry", "value":"Account.Industry", "type":"ra_picklist2", "sortable":true, "editable":true, "initialWidth":160, "iconName":"utility:company",
"typeAttributes": {"rowId": { "fieldName": "record.Id" },"objectDotField": "Account.Industry","parentRecordId": "AccountId"}},

{"label":"Acc: Customer Priority", "fieldName":"record.Account.CustomerPriority__c", "value":"Account.CustomerPriority__c", "type":"ra_picklist2", "sortable":true, "editable":true, "initialWidth":160, "iconName":"utility:company",
"typeAttributes": {"rowId": { "fieldName": "record.Id" },"objectDotField": "Account.CustomerPriority__c","parentRecordId": "AccountId"}},

{"label":"Customer Priority","fieldName":"record.Account.CustomerPriority__c","value":"Account.CustomerPriority__c","type":"ra_picklist","typeAttributes": {"rowId": { "fieldName": "record.Id" },"objectDotField": "Account.CustomerPriority__c"},"editable":true,"sortable":true}



Example 2

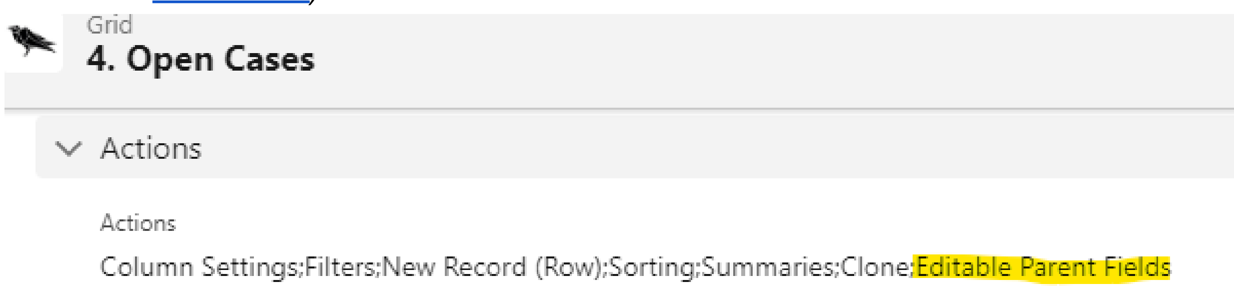
Or for a custom parent object, project__c, to get a field called start_date__c use
project__r.start_date__c:
SELECT Id, Name, project__r.start_date__c FROM Opportunity

You can also make parent fields editable, see more details [here]

Editable Parent Fields
Grids allow you to edit parent fields on a grid. We have a dedicated Action for this called 'Editable Parent Fields'.

Standard field types
To enable parent field edit for standard field types simply follow the steps below:

1. On the grid record enable the action 'Editable Parent Fields' *(if you don't see this option see [below in red])*:

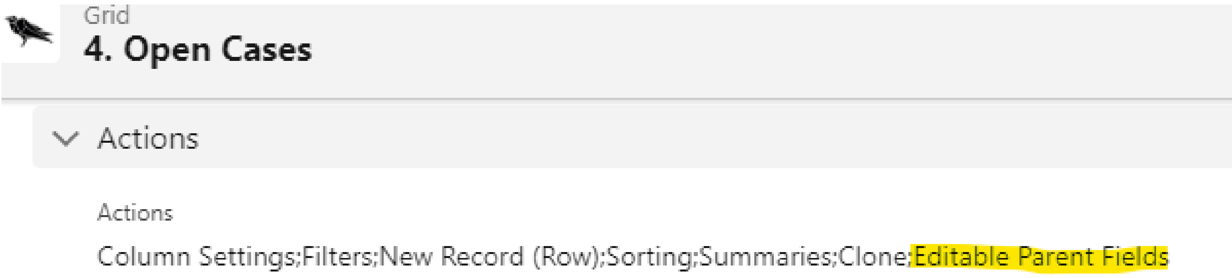


2. Add the parent field to the grid that you wish to edit using the [standard Salesforce parent field notation]. E.g on an opportunity grid to edit the parent account's employee use:
   a. *{"label":"Account: Employees", "fieldName":"record.Account.NumberOfEmployees", "value":"Account.NumberOfEmployees","type":"number","sortable":true,"editable":true, "initialWidth": 200}*
3. Add the parent field to the SOQL query:
   a. E.g. from above example add in *Account.NumberOfEmployees*

If a user tries to edit a parent field without having completed step 1 they will get an error message "No such column 'Account.AnnualRevenue' on sobject of type Contact - (System Code)"

Custom field types
To enable parent field edit for standard field types simply follow the steps below:

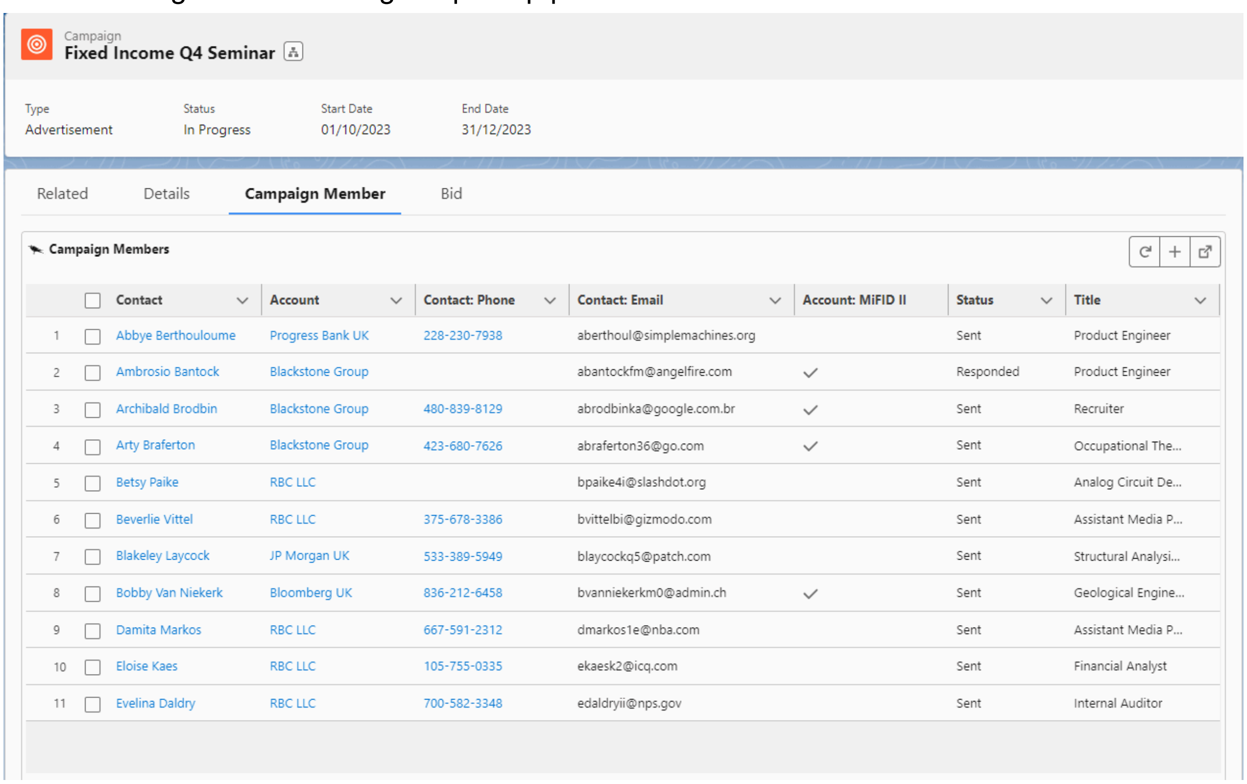
1. On the grid record enable the action 'Editable Parent Fields' *(if you don't see this option see [below in red])*:



2. Add the parent field to the grid that you wish to edit using the [standard Salesforce parent field notation]() and adding in a "field". E.g on an opportunity grid to edit the parent account's employee use:
    a. *{"label":"Account: Industry", "fieldName":"record.Account.Industry","value":"Account.Industry","type":"ra_picklist","sortable":true,"editable":true, "initialWidth": 200, "typeAttributes": {"rowId": { "fieldName": "record.Id" },"objectDotField": "Account.Industry", "field":"record.Account.Industry"}}*
3. Add the parent field to the SOQL query:
    a. E.g. from above example add in *Account.Industry*

Example

On the Campaign Member object you may wish to edit parent Contacts email and phone details without having the user having to open up parent contact records



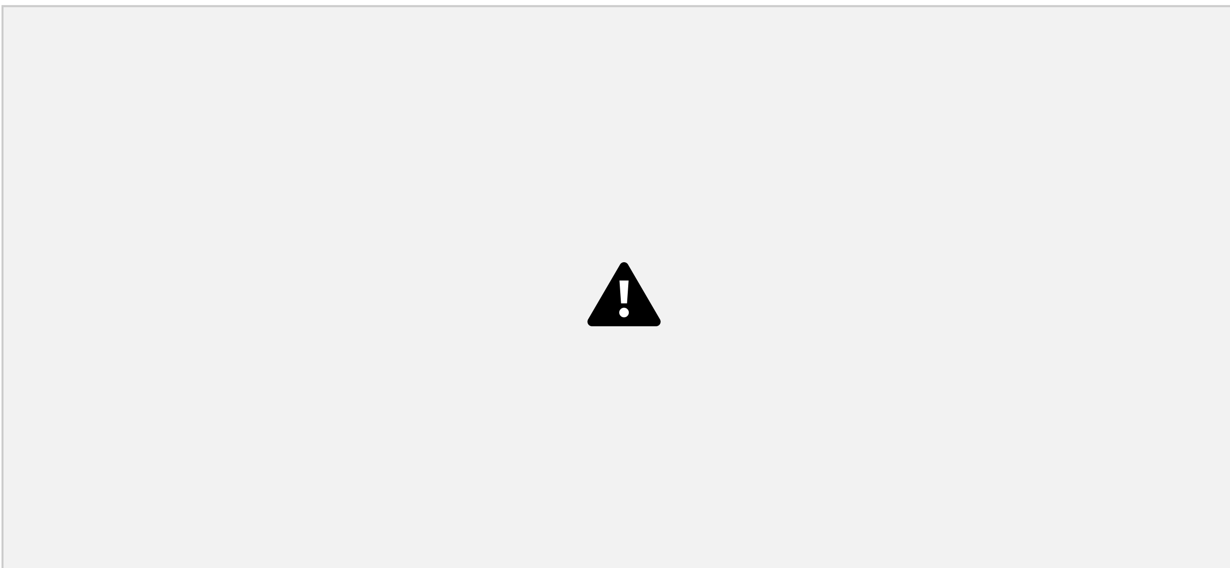
	Contact	Account	Contact: Phone	Contact: Email	Account: MiFID II	Status	Title
1	Abbye Berthouloume	Progress Bank UK	228-230-7938	aberthoul@simplemachines.org		Sent	Product Engineer
2	Ambrosio Bantock	Blackstone Group		abantockfm@angelfire.com	✓	Responded	Product Engineer
3	Archibald Brodbin	Blackstone Group	480-839-8129	abrodbinka@google.com.br	✓	Sent	Recruiter
4	Arty Braferton	Blackstone Group	423-680-7626	abraferton36@go.com	✓	Sent	Occupational The...
5	Betsy Paike	RBC LLC		bpaike4i@slashdot.org		Sent	Analog Circuit De...
6	Beverlie Vittel	RBC LLC	375-678-3386	bvittelbi@gizmodo.com		Sent	Assistant Media P...
7	Blakeley Laycock	JP Morgan UK	533-389-5949	blaycockq5@patch.com		Sent	Structural Analysi...
8	Bobby Van Niekerk	Bloomberg UK	836-212-6458	bvanniekerkm0@admin.ch	✓	Sent	Geological Engine...
9	Damita Markos	RBC LLC	667-591-2312	dmarkos1e@nba.com		Sent	Assistant Media P...
10	Eloise Kaes	RBC LLC	105-755-0335	ekaesk2@icq.com		Sent	Financial Analyst
11	Evelina Daldry	RBC LLC	700-582-3348	edaldryii@nps.gov		Sent	Internal Auditor

Invoke Flows - **Paid**

You can create flows to run off Grids by building your flow as normal in Salesforce and then referencing it in the Grid.

Creating Mass / Header Actions using Flows

The Custom Header/Mass Actions allows you to create flows based off the rows selected in the grid. See example below:



To implement this populate the Grid records ‘**Custom Header/Mass Actions**’ field like example below;

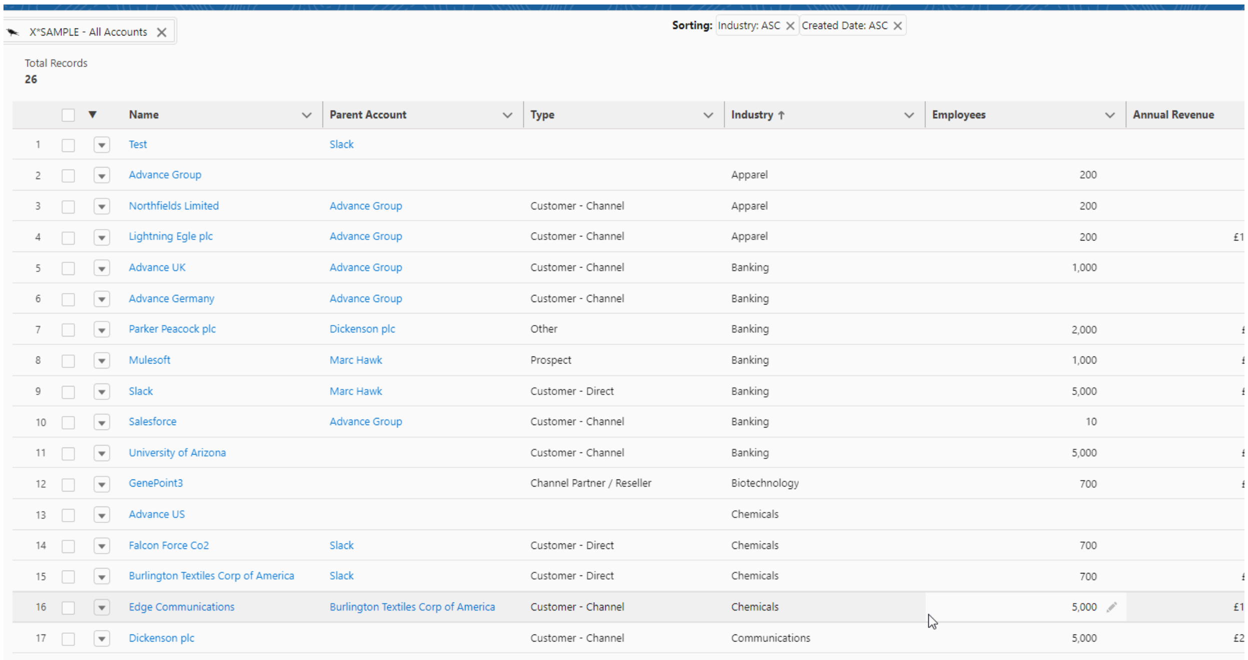
{"label":" Commercial & Technical QA","flowApiName":"CommercialTechQA","params":"ids"}}

The flow needs to include an ‘ids’ text variable with ‘Available for input’ checked.

Custom Row Action

The Custom Row action must be populated as below where ‘Create_Contact’ is the Flow API name:

{"label": "New Contact", "name": "invoke_flow","flowApiName":"Create_Contact", "params":"ids"}



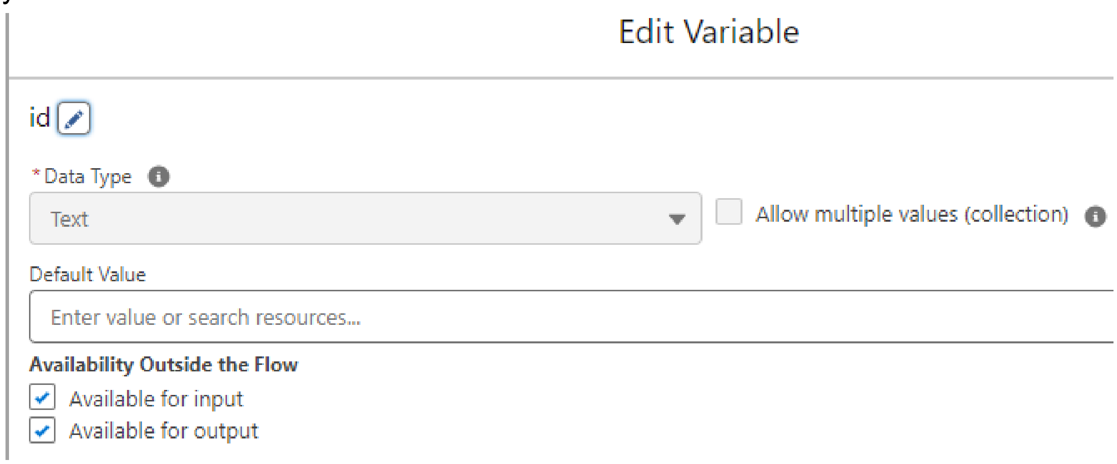
X*SAMPLE - All Accounts

Sorting: Industry: ASC Created Date: ASC

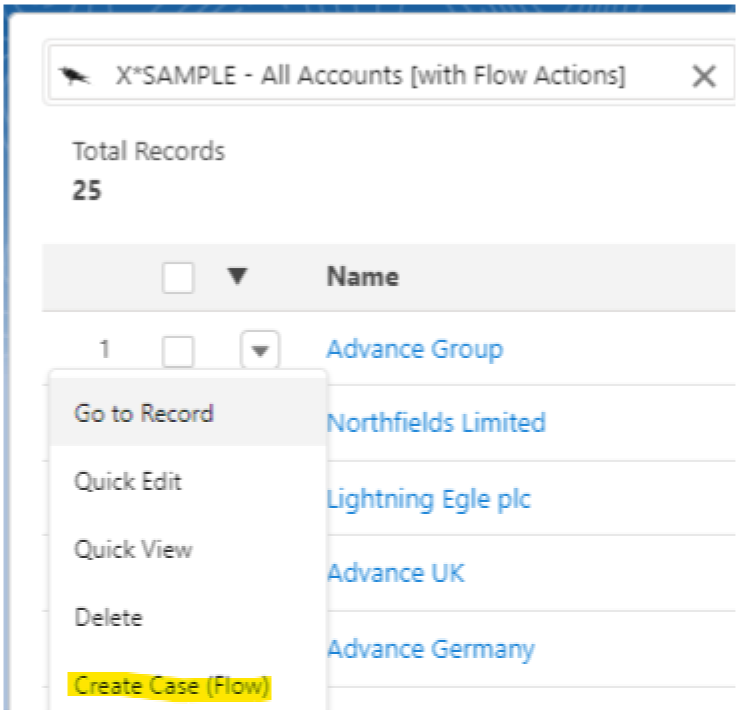
Total Records
26

	Name	Parent Account	Type	Industry ↑	Employees	Annual Revenue
1	Test	Slack				
2	Advance Group			Apparel	200	
3	Northfields Limited	Advance Group	Customer - Channel	Apparel	200	
4	Lightning Egle plc	Advance Group	Customer - Channel	Apparel	200	£1
5	Advance UK	Advance Group	Customer - Channel	Banking	1,000	
6	Advance Germany	Advance Group	Customer - Channel	Banking		
7	Parker Peacock plc	Dickenson plc	Other	Banking	2,000	£
8	Mulesoft	Marc Hawk	Prospect	Banking	1,000	£
9	Slack	Marc Hawk	Customer - Direct	Banking	5,000	£
10	Salesforce	Advance Group	Customer - Channel	Banking	10	
11	University of Arizona		Customer - Channel	Banking	5,000	£
12	GenePoint3		Channel Partner / Reseller	Biotechnology	700	£
13	Advance US			Chemicals		
14	Falcon Force Co2	Slack	Customer - Direct	Chemicals	700	
15	Burlington Textiles Corp of America	Slack	Customer - Direct	Chemicals	700	£
16	Edge Communications	Burlington Textiles Corp of America	Customer - Channel	Chemicals	5,000	£1
17	Dickenson plc		Customer - Channel	Communications	5,000	£2

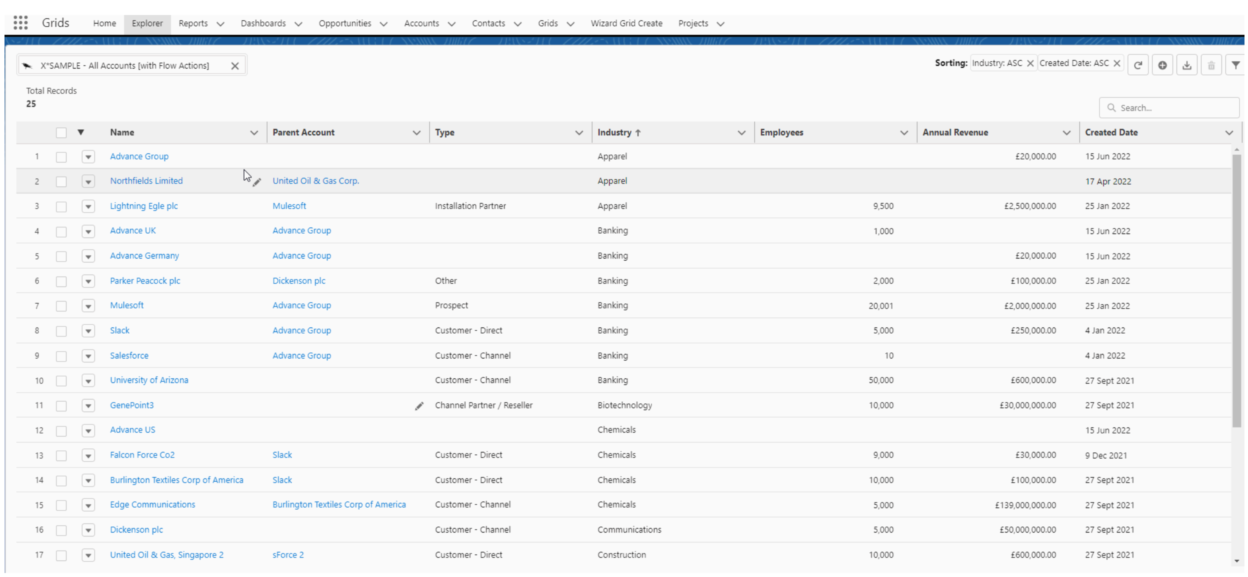
Remember to ensure the flow above has both a contextId and Ids variables as mentioned here, you will also need a id variable to reference the row the flow was launched from:



The button then appears on the record drop downs:

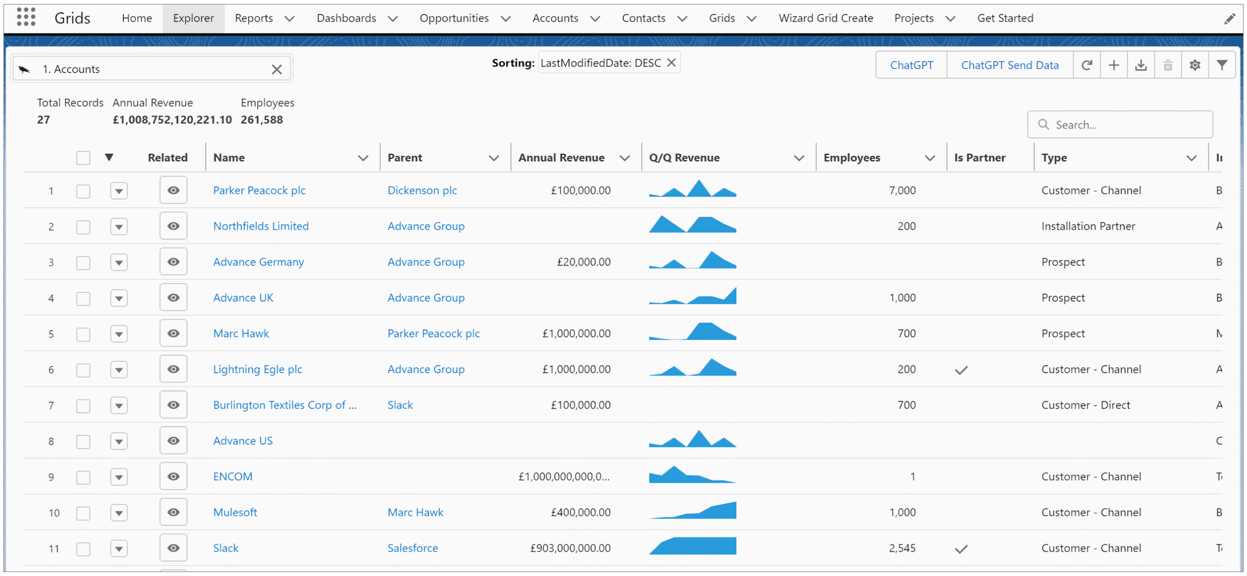


Here is an example gif showing a screen flow being started from a grid that create a case on an Account:



Column Settings

If you wish to grant your users the ability to update grid columns themselves then you can do this using the ‘Column Settings’ action on the grid. The user then clicks on the settings tab on the top right of the grid to dynamically add, remove and move columns in the grid:



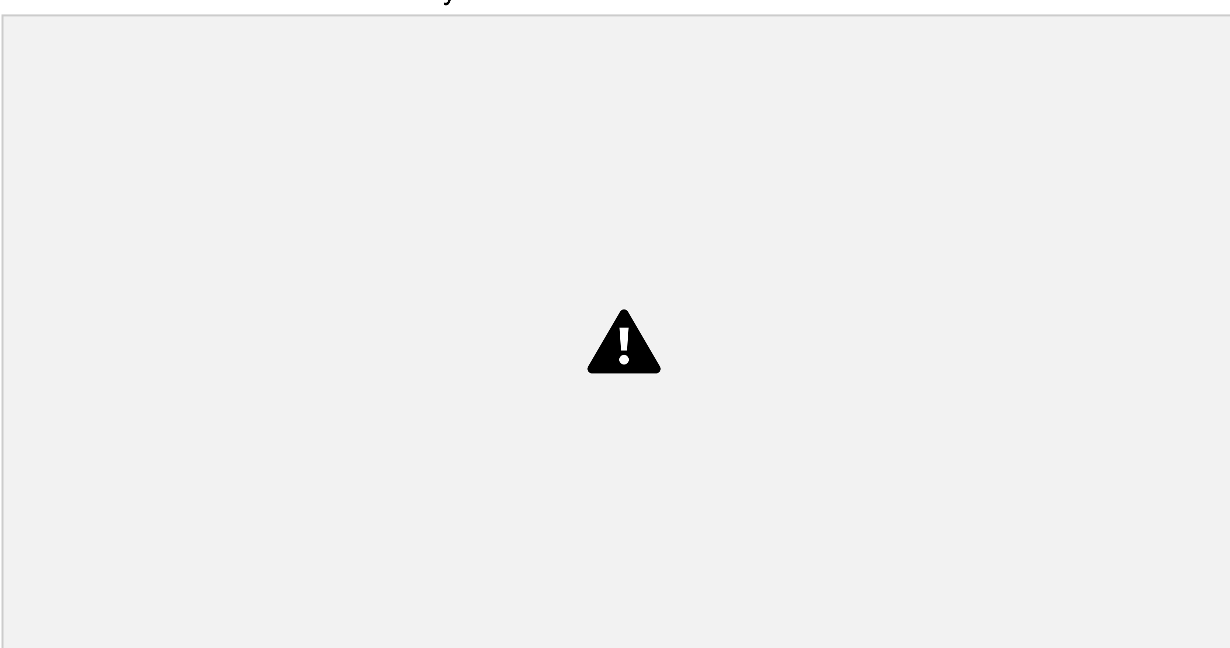
This functionality is dynamic for each user and resets on refresh of the page or the grid.

To enable Column Settings on a grid simply:

- On the Grid record click on the action field and add the 'Column Settings' action (if you do not see this option then please see below).

Clone Rows

Allow users to clone Grids Rows by added the Clone button to the Grids:



Please note this functionality clones the visible data on the row it does not clone other fields. If you would like other fields to also be cloned then please reach out to us at info@raven-apps.com

To enable Clone rows on a grid simply:

- On the Grid record click on the action field and add the ‘Clone’ action (if you do not see this option then please see below).

Clone exclusion field

If you wish to clone rows but exclude certain fields you can do this through using the ‘Clone Exclusion Fields’ field on the Grid (as below in red ensure you have the latest package version for this).

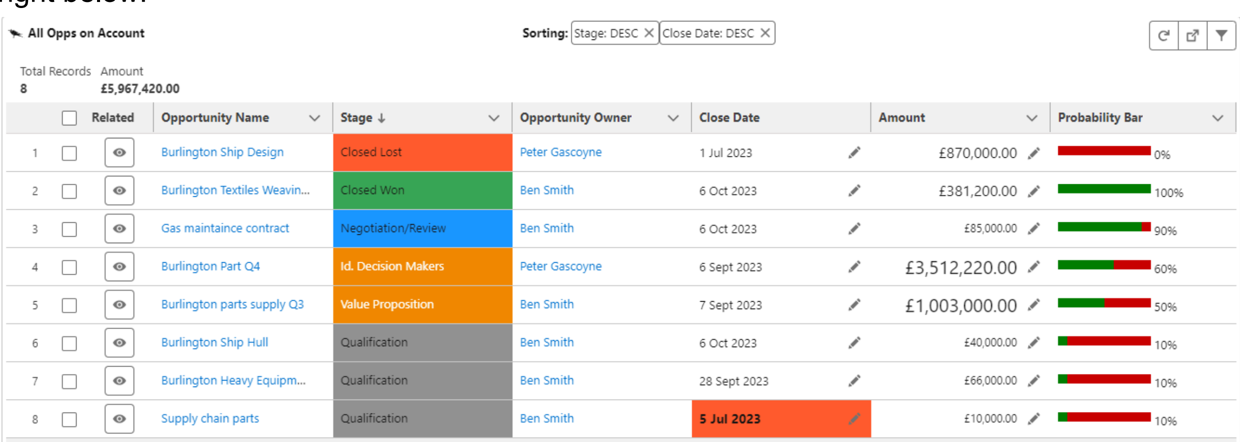
Formula Image Columns

Example 1

If you wish to include a completion bar in your Grid you can do this by

1. Create a formula field (Probability_Bar__c)
    a. *IMAGE("/img/samples/color_green.gif","green",10,100*Probability) & IMAGE("/img/samples/color_red.gif","red",10, 100-100*Probability) & " " & TEXT(100*Probability) & "%"*
2. Add a Column for the completion bar
    a. *{"label":"Probability Bar", "fieldName":"record.Probability_Bar__c", "value":"Probability_Bar__c", "type":"ra_formulaImage", "sortable":false, "initialWidth": 200, "editable":false, "typeAttributes": {"rowId": { "fieldName": "record.Id"}, "field":"Probability_Bar__c"}}*

adding a formula field to object and then adding a column, this appears as the column on the right below:



In the above:

- ‘10’ in each image indicates the height of the bar, if you wish to have say the red section thinner then update the value to 5 in the second image
- The colors defines the color of the bar, you can update to any of the below:
    - /img/samples/color_red.gif
    - /img/samples/color_yellow.gif
    - /img/samples/color_green.gif
- The field is multiplied by 100 to extend the bar to full width, you can reduce this number if you want the bar to be smaller horizontally.
- The TEXT part at the end is optional to include the text value of the amount.

Example 2

1. Create a formula field (Probability_Bar__c)

    IF(NumberOfEmployees <100,IMAGE("/img/samples/color_red.gif","red",10,NumberOfEmployees/100),
    IF(NumberOfEmployees <1000, IMAGE("/img/samples/color_yellow.gif","yellow",10,NumberOfEmployees/100),

    IMAGE("/img/samples/color_green.gif","green",10,NumberOfEmployees/100)))

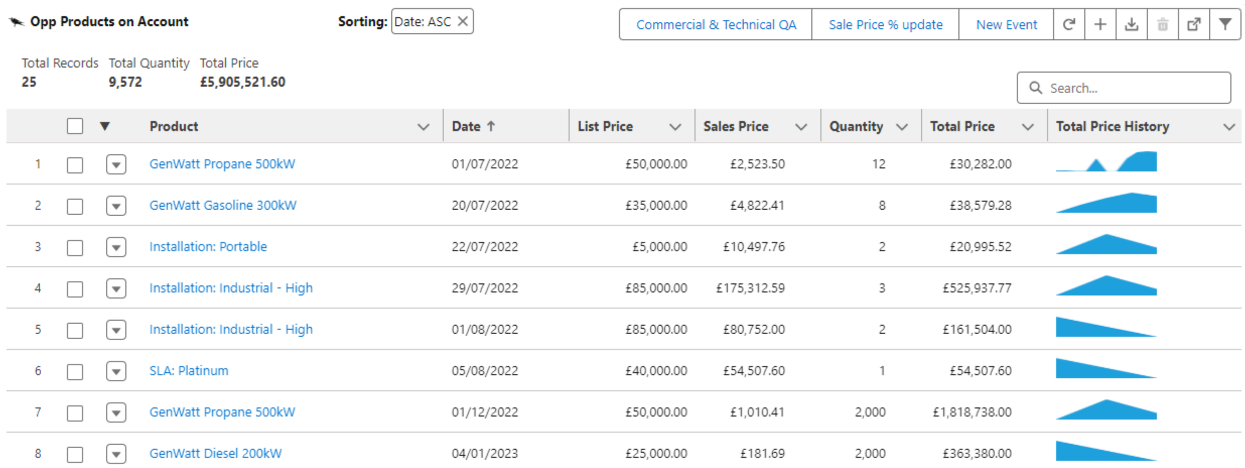
2. Add a Column for the completion bar
    a. *{"label":"Employees", "fieldName":"record.Employees_Bar__c", "value":"Employees_Bar__c", "type":"ra_formulaImage", "sortable":false, "initialWidth": 200, "editable":false, "typeAttributes": {"rowId": { "fieldName": "record.Id"}, "field":"Employees_Bar__c"}}*

Name	# of Opps ↓	Value of Opps	Employees
United Oil & Gas Corp.	12	7,728,001	
Burlington Textiles Corp of ...	8	5,968,420	
Grand Hotels & Resorts Ltd	5	860,000	
Edge Communications	5	1,182,000	
Falcon Force Co2	4	131,257	
Apple	4	7,000	
GenePoint3	3	175,000	
University of Arizona	3	240,000	
Express Logistics and Trans...	3	420,000	

Sparklines

Sparklines can be created to show the trend of a value on a record over time.

E.g. in the example below the Total Price changes over time are represented to show trends / negotiation status.

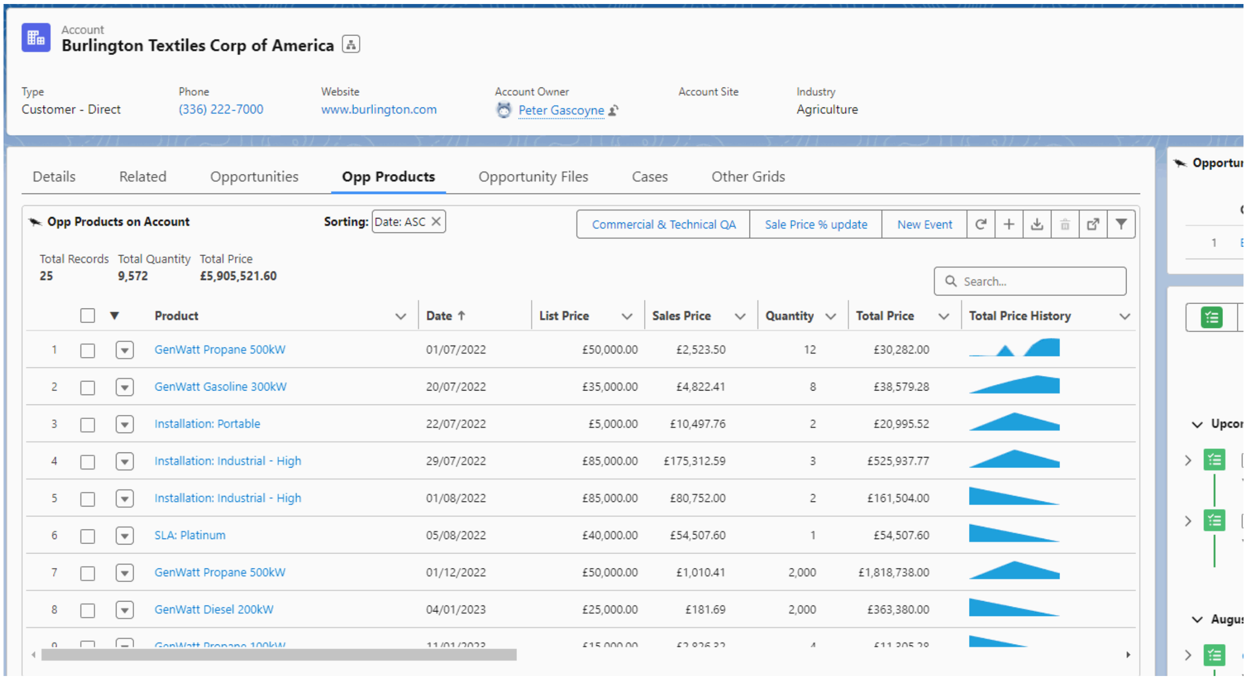


	Product	Date	List Price	Sales Price	Quantity	Total Price	Total Price History
1	GenWatt Propane 500kW	01/07/2022	£50,000.00	£2,523.50	12	£30,282.00	
2	GenWatt Gasoline 300kW	20/07/2022	£35,000.00	£4,822.41	8	£38,579.28	
3	Installation: Portable	22/07/2022	£5,000.00	£10,497.76	2	£20,995.52	
4	Installation: Industrial - High	29/07/2022	£85,000.00	£175,312.59	3	£525,937.77	
5	Installation: Industrial - High	01/08/2022	£85,000.00	£80,752.00	2	£161,504.00	
6	SLA: Platinum	05/08/2022	£40,000.00	£54,507.60	1	£54,507.60	
7	GenWatt Propane 500kW	01/12/2022	£50,000.00	£1,010.41	2,000	£1,818,738.00	
8	GenWatt Diesel 200kW	04/01/2023	£25,000.00	£181.69	2,000	£363,380.00	

To create a Sparkline column you will need to:

1. Create a text field on the object that stores historic changes
    a. E.g. Price_History__c
2. Create a formula field references the above field
    a. E.g. Price_History_Sparkline__c
    b. *IMAGE('https://quickchart.io/chart?c={ type: "sparkline", data: { datasets: [{backgroundColor: "rgb(33, 160, 223)", data: ['+Price_History__c+']}] }}', 'score progression', 20, 100)*
3. Add a JSON column into the Grid which references the formula field:
    a. *{"label":"Total Price History", "fieldName":"record.Price_History_Sparkline__c", "value":"Price_History_Sparkline__c", "type":"ra_formulaImage", "sortable":false, "initialWidth": 200, "editable":false, "typeAttributes": {"rowId": { "fieldName": "record.Id"}, "field":"Price_History_Sparkline__c"}}*
4. Create a before flow (or similar) to update the text field to store the latest value, you can see an example [here](#)

You can also add a flow to mass update the values on selected rows - see the [Creating Flows on Grids](#) section for more details.

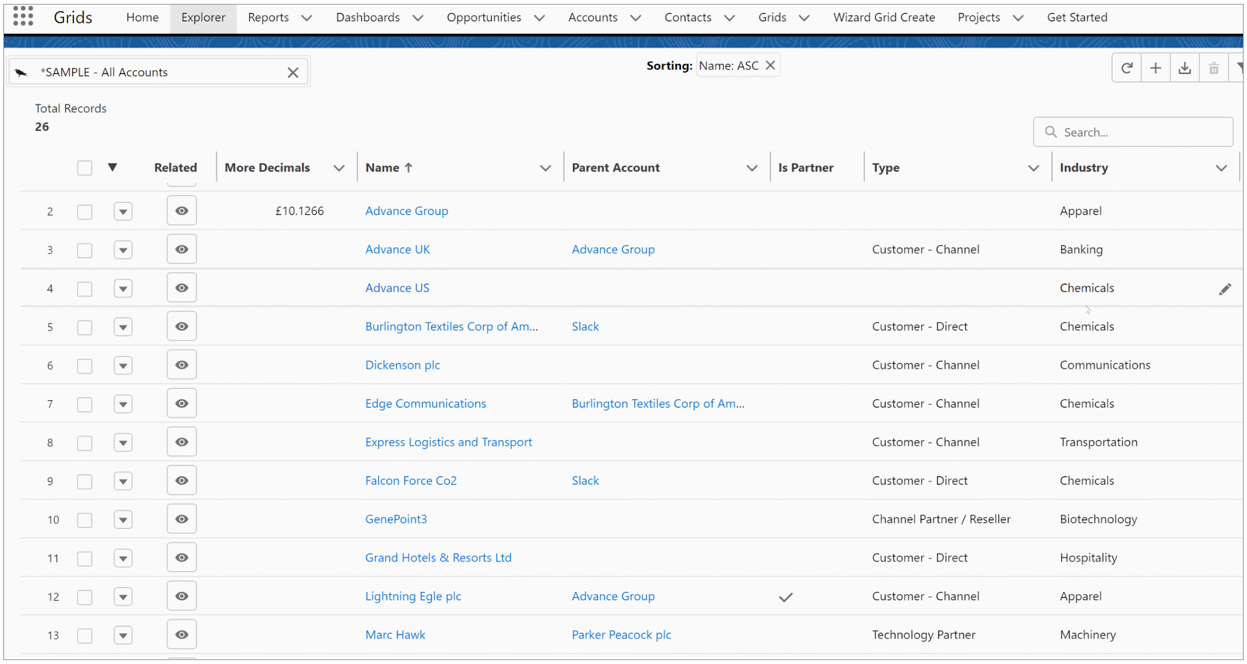


The flow that is used on the above is [here]() and is called 'Opp_Product_Sale_Price_update'. This flow button is put on the grid by populating the 'Custom Header/Mass Action' field with:
{"label":"Sale Price % update","flowApiName":"Opp_Product_Sale_Price_update", "params":"ids"}

Related Grids

Having grids open within other grids can be done by utilizing the 'Related Grids'. Simply reference the grids you wish to link to the existing grid, e.g.
{"gridId":"a008d000003fA5PAAU","sectionLabel":"Contacts"},
{"gridId":"a008d000003fA5eAAE","sectionLabel":"Opportunities"}

You will also have to add a column of type "button-icon" with JSON as below:
{"label":"Related", "fieldName":"related", "type":"button-icon","initialWidth":75,"value":"Related","typeAttributes":{"name":"ra_viewRelated","iconName":"action:preview","variant":"border-filled","alternativeText":"View Related"}}

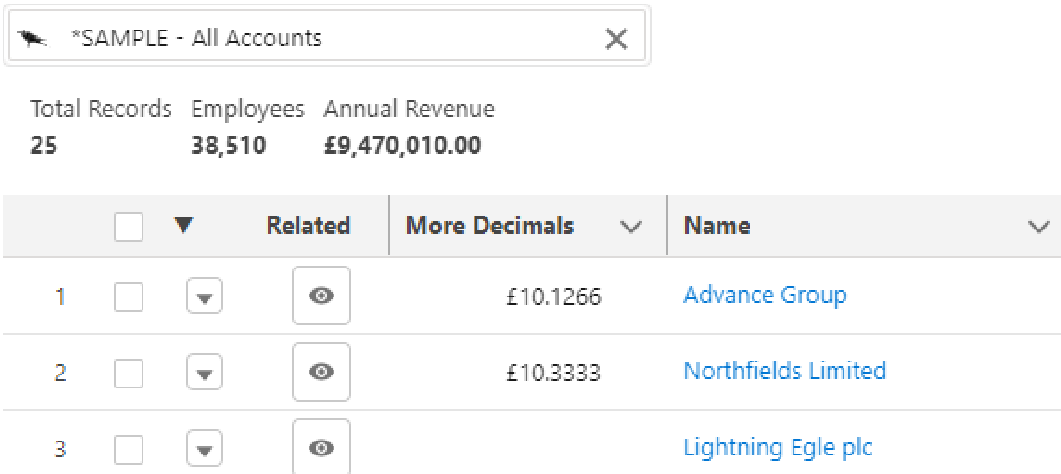


	Related	More Decimals	Name	Parent Account	Is Partner	Type	Industry
2		£10.1266	Advance Group				Apparel
3			Advance UK	Advance Group		Customer - Channel	Banking
4			Advance US				Chemicals
5			Burlington Textiles Corp of Am...	Slack		Customer - Direct	Chemicals
6			Dickenson plc			Customer - Channel	Communications
7			Edge Communications	Burlington Textiles Corp of Am...		Customer - Channel	Chemicals
8			Express Logistics and Transport			Customer - Channel	Transportation
9			Falcon Force Co2	Slack		Customer - Direct	Chemicals
10			GenePoint3			Channel Partner / Reseller	Biotechnology
11			Grand Hotels & Resorts Ltd			Customer - Direct	Hospitality
12			Lightning Egle plc	Advance Group	✓	Customer - Channel	Apparel
13			Marc Hawk	Parker Peacock plc		Technology Partner	Machinery

Ensure that the grids you are referencing have a variable which relates them to the parent record. E.g. for the above Contact related grid the query is
SELECT Id, FirstName, LastName, Title, AccountId, Account.Name, Email, Phone, Department FROM Contact WHERE Accountid =: sv1
And the Variables is:
{"type":"record", "field":"id", "variable":"sv1"}

Decimal Places

Columns
To manage decimal places in columns use the JSON format:
{"label":"Price",
"fieldName":"record.Price__c","value":"Price__c","type":"currency","sortable":true,"editable":true,
"initialWidth": 150,
"typeAttributes": { typeAttributes: { step: “0.0001”, minimumFractionDigits: null, maximumFractionDigits: 4}}
The above example would display 4 decimal places using maximumFractionDigits and the step for inline edit (you will need the Salesforce field to have 4 decimal points set in the field definition too).



Summaries
To manage decimal places in summaries use the JSON format:
"maximumFractionDigits": "0"
The above example would display 0 decimal places

E.g.:
{"label": "Annual Revenue", "field":"AnnualRevenue", "type": "SUM", "formatStyle": "currency","maximumFractionDigits": "0"}

Smart Links

When you click a hyperlink on a grid by default it will open the record in a new browser tab. If you wish to open in a different way you can add the below to a column:

JSON	Outcome on click:
"target":"_self"	Opens a new workspace tab or subtab in console apps, depending on the user's context. Opens in the same tab in apps with standard navigation.
"target":"_workspaceTab"	Opens a new workspace tab in console apps. Opens a new browser tab in apps with standard navigation.
"target":"_subtab"	Opens a new tab in console apps. Opens in the same tab in apps with standard navigation.

For more details see Salesforce documentation here.

Smart links work for both ra_name and ra_lookup field type.

Example JSON for a column would be:
{"label":"Quote", "fieldName":"record.Name", "value":"Name", "type":"ra_name", "editable":false, "sortable":true, "wrapText":true, "typeAttributes": {"rowId": { "fieldName": "record.Id" }, "target":"_self"},"initialWidth":300}

Dynamic Grids

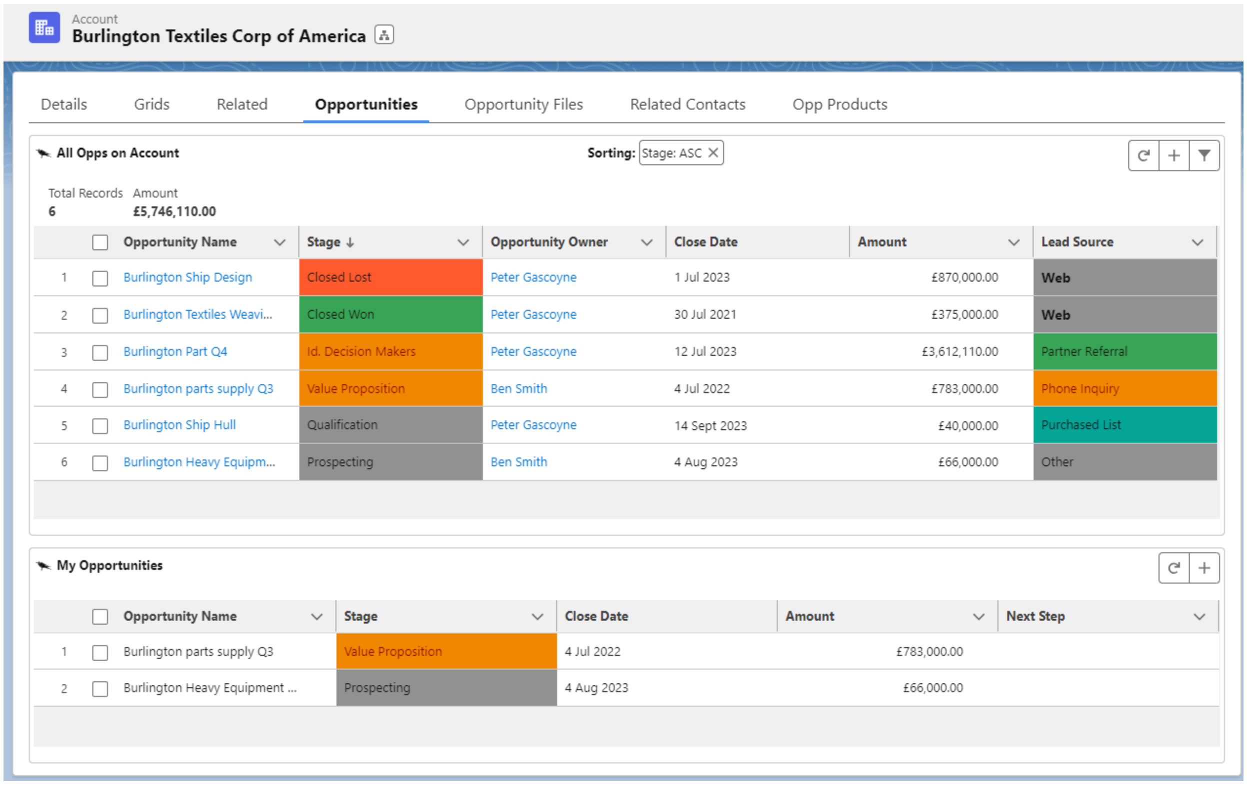
You may want a grid to show the user values specific to them, i.e. dynamic grids. This can be done by adding at the end of the query *USING SCOPE mine*
For example if you wanted a Opportunity grid to only show the user their Opportunities you could use the below query:
SELECT Name, Owner.Name, Account.Name, Amount, ExpectedRevenue, CloseDate, Probability, StageName FROM Opportunity USING SCOPE mine

You can also show your teams, your territories, delegated records etc… by replacing 'mine' above with one of the options below:

- delegated: Records delegated to other users. For example, a task that we have delegated to another person.
- everything: All records. It is the same as not including the SCOPE clause.
- mine: Only the records of the user who is executing the query.
- mine_and_my_groups: This filter only applies to the ProcessInstanceWorkItem object of the Approval Process. It will return only the records assigned to us or to one of the members.
- my_territory: It will only be available if we have territory management enabled. It will show us only the records of the territory of the user that is executing the query.
- my_team_territory: It will only be available if we have territory management enabled. It will show us only the records of the territory of the user that is executing the query or of one of the user's team.
- scopingRule: It will only be available if we have activated some Scoping Rule on the object that we are consulting. It will only show us the records defined in the Scoping Rule that applies to the user that is executing the query.
- team: This Grid shows all records owned by you or owned by users below you in the role hierarchy.

More details on advanced queries can be found [here]

E.g. The below example has the 'All Opps on Account' Grid with all 6 Opps and the second grid 'My Opportunities' shows just the current user's Opportunities.

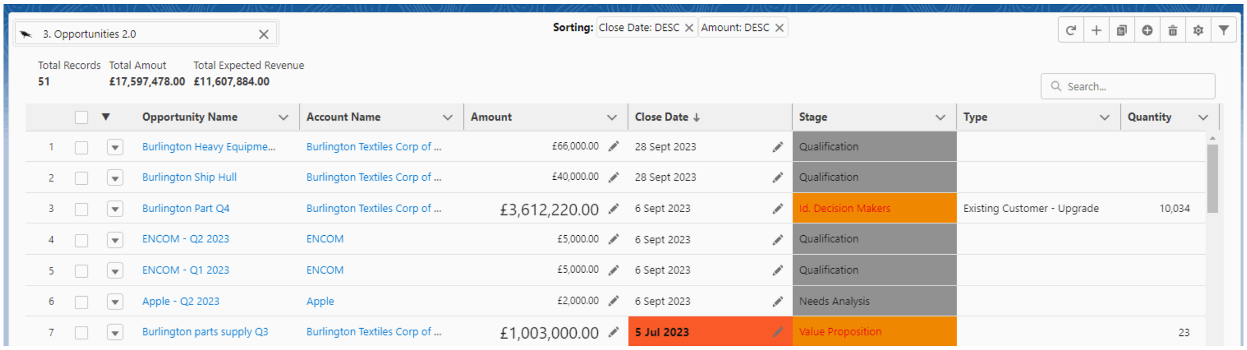


Default values for inline create

If you wish to set a default value in a Grid then you can populate the field 'New Record Row Defaults' with the field and value you want the grid to default with. The format for this is:
{"field":"Type", "type": "value", "value":"New Customer"}

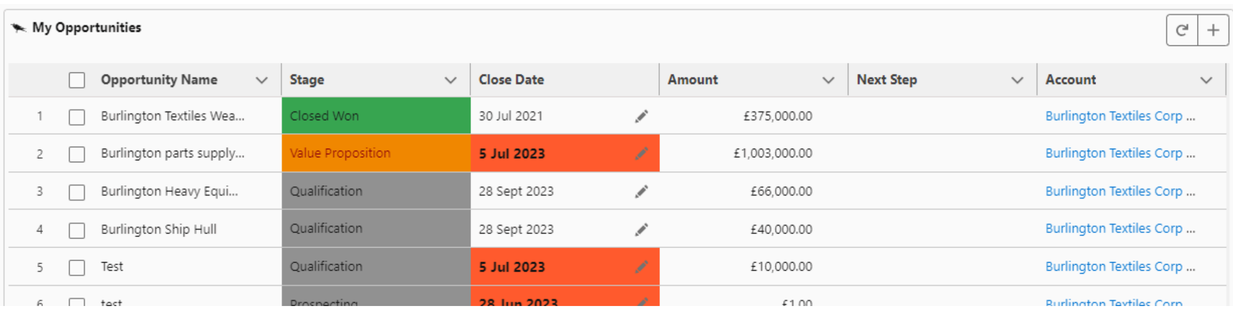
If a field is defaulted within Salesforce standard functionality it will default to this value if it is not on the grid otherwise you should add a 'New Record Row Defaults' as above. If a field is required and has a default you do not need to add it to the grid to create the record, it will default on inline create save.

Example
Having the below in the 'New Record Row Defaults' fields will make these 4 values default as in below gif
{"field":"Type", "type": "value", "value":"New Customer"},
{"field":"TotalOpportunityQuantity", "type": "value", "value":1},
{"field":"Amount", "type": "value", "value":10000},
{"field":"StageName", "type": "value", "value":"Qualification"}



For Related List Grids the parent default field can also be automated, this will be done automatically when using the Grid builder. For reference it will populated the ‘New Record Row Defaults’ with 2 variables that refer to the parent record, e.g. if the Account is the parent it will use:
{"field":"AccountId", "type": "variable", "value":"sv1"},
{"field":"Account.Name", "type": "variable", "value":"sv2"}

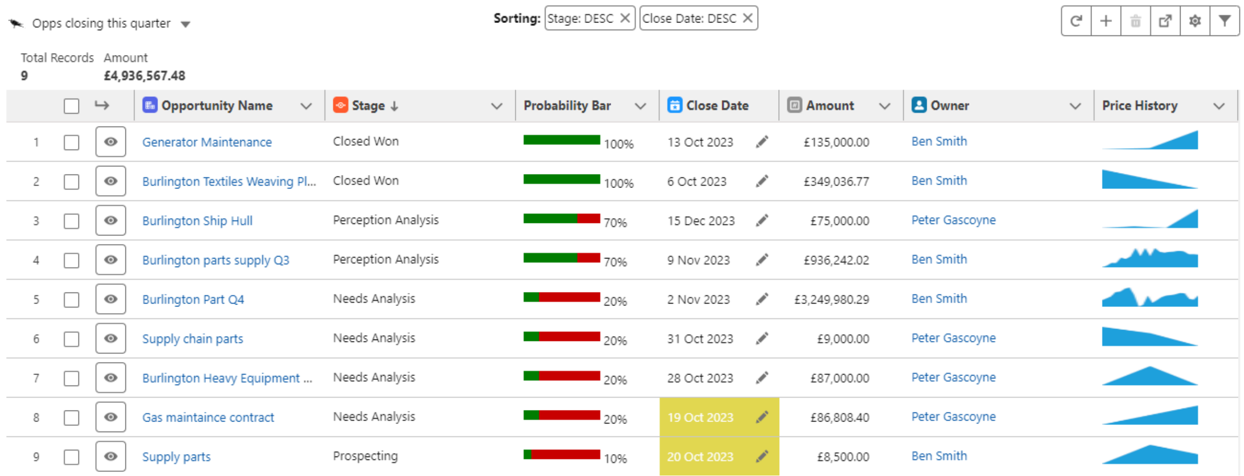
Example:



Field Header Icons

You can replace column headers with icons using the [icon library here](). You just need to add the below into the JSON (here the icon favorite is used)
"iconName": "utility:favorite"

Example 1
You can use field header icons to indicate which columns are editable to end users



Opps closing this quarter

Sorting: Stage: DESC Close Date: DESC

Total Records 9

Amount £4,936,567.48

	Opportunity Name	Stage	Probability Bar	Close Date	Amount	Owner	Price History
1	Generator Maintenance	Closed Won	100%	13 Oct 2023	£135,000.00	Ben Smith	
2	Burlington Textiles Weaving Pl...	Closed Won	100%	6 Oct 2023	£349,036.77	Ben Smith	
3	Burlington Ship Hull	Perception Analysis	70%	15 Dec 2023	£75,000.00	Peter Gascoyne	
4	Burlington parts supply Q3	Perception Analysis	70%	9 Nov 2023	£936,242.02	Ben Smith	
5	Burlington Part Q4	Needs Analysis	20%	2 Nov 2023	£3,249,980.29	Ben Smith	
6	Supply chain parts	Needs Analysis	20%	31 Oct 2023	£9,000.00	Peter Gascoyne	
7	Burlington Heavy Equipment ...	Needs Analysis	20%	28 Oct 2023	£87,000.00	Peter Gascoyne	
8	Gas maintaince contract	Needs Analysis	20%	19 Oct 2023	£86,808.40	Peter Gascoyne	
9	Supply parts	Prospecting	10%	20 Oct 2023	£8,500.00	Ben Smith	

The JSON for the above has the below added in for each column:
Related Grids column: *"iconName": "utility:level_down"*
Opportunity Name column: *"iconName": "standard:opportunity"*
Stage column: *"iconName": "standard:stage"*
Close Date column: *"iconName": "standard:key_dates"*
Amount column: *"iconName": "standard:number_input"*
Owner column: *"iconName": "standard:user"*

Example 2
You can use field header icons to replace field names:

		User Story Name
1	✓	Add Ids Object Type Attribute to Flow
2	✓	Edit Parent Fields

The JSON column for the above is below simply required adding the iconName and leaving the label blank:
{"label":"","fieldName":"record.Focus__c","value":"Focus__c","type":"boolean","sortable":true,"initialWidth":70,"editable":true, "iconName": "utility:favorite"}

If you populate the label and the iconName you will get both the icon and the label as in Example 1.

Image Cells

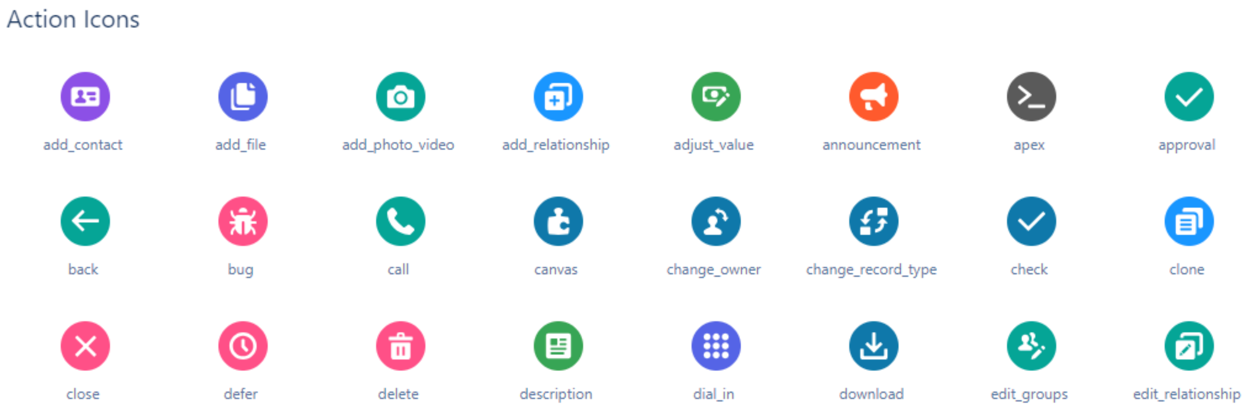
There are 4 ways to display images in formula fields, we **recommend using the first method** but have included all options for completeness:

1. **Reference the Lightning Design System Icons**

2. Using Salesforce Sample images
3. Using Salesforce Graphics Pack from AppExchange
4. Loading Custom Images

1. Reference the Lightning Design System Icons

The Lightning Design System Icons give ~1,000 icons that can be used on your Grids:



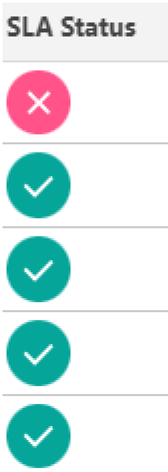
To create dynamic image cell using the LDS Icons follow the below 3 steps:

1. Create a formula field (text) *IMAGE_FORMULA_FIELD* using the Lightning Design System Icons and the format Icon_Area:Icon_Name, e.g.
    a. *IF (CONDITION STATEMENT, "Icon_Area:Icon_Name", "Icon_Area:Icon_Name")*
2. Add this formula field as a column:
    a. *{"label":"LABEL", "type":"text", "sortable":true,"cellAttributes": {"iconName":{"fieldName": "record.IMAGE_FORMULA_FIELD"}}},*
3. Include the formula field *IMAGE_FORMULA_FIELD* in the SOQL query on the Grid record.

Example

1. Created formula field '*SLA_Status__c*':
    a. *IF(Planned_Date__c <NOW(), "action:close", "action:approval")*
2. Create JSON Column:
    a. *{"label":"SLA Status", "type":"text", "sortable":true,"cellAttributes": {"iconName":{"fieldName": "record.SLA_Status__c"}}}*
3. Include the formula field *IMAGE_FORMULA_FIELD* in the SOQL query on the Grid record.

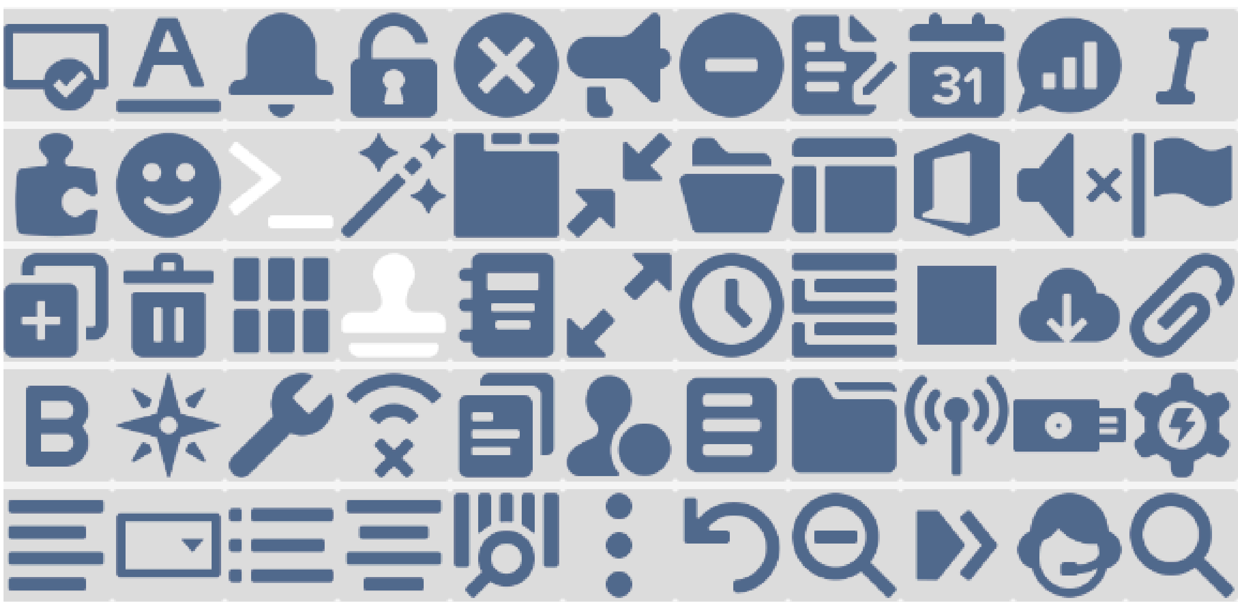
This gives the SLA status column as below:



2. Salesforce Sample Images

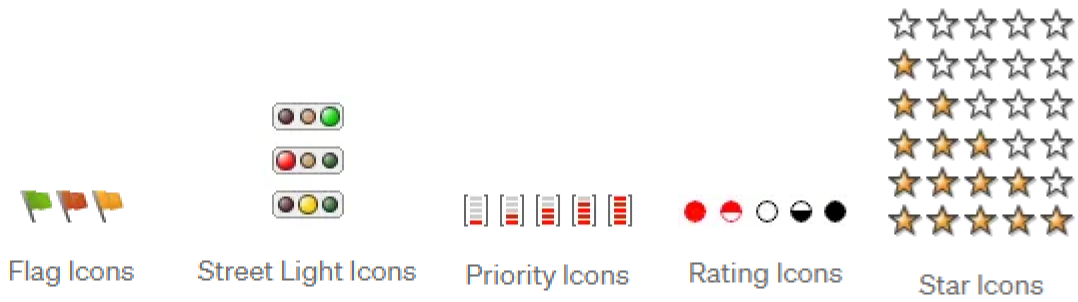
You can reference Salesforce sample images:

- Lightning Images can be found here (at bottom):



:

- Classic images are here, e.g.:



To create create a dynamic image cell using Salesforce sample images follow the 3 steps below:

1. Create a formula field using the referenced images above, e.g.
   a. *IF(CONDITION STATEMENT, IMAGE("/img/icon/t4v35/utility/close_120.png","SLA", 16,16), IMAGE("/img/icon/t4v35/utility/check_120.png","SLA", 16,16))*
2. Add this formula field as a column:
   a. *{"label":"LABEL", "fieldName":"record.IMAGE_FORMULA_FIELD", "value":"IMAGE_FORMULA_FIELD", "type":"ra_formulaImage", "sortable":false, "initialWidth": 200, "editable":false, "typeAttributes": {"rowId": { "fieldName": "record.Id"}, "field":"IMAGE_FORMULA_FIELD"}}*
3. Include the formula field *IMAGE_FORMULA_FIELD* in the SOQL query on the Grid record.

Example

1. Create formula field '*SLA_Status__c*' from the images here:
   a. *IF(Planned_Date__c <NOW(), IMAGE("/img/icon/t4v35/utility/close_120.png","SLA", 16,16), IMAGE("/img/icon/t4v35/utility/check_120.png","SLA", 16,16))*
2. Created the column in the grid record:
   a. *{"label":"SLA Status", "fieldName":"record.SLA_Status__c", "value":"SLA_Status__c", "type":"ra_formulaImage", "sortable":false, "initialWidth": 200, "editable":false, "typeAttributes": {"rowId": { "fieldName": "record.Id"}, "field":"SLA_Status__c"}},*
3. Add the field *SLA_Status__c* into the SOQL query

This gives the SLA status column as below:

3. Salesforce Graphics Pack

Salesforce Labs has provided a Graphics Pack that you can download for free [here](#).

To create create a dynamic image cell using Salesforce sample images follow the 3 steps below:

1. Create a formula field using the referenced images above, e.g.
   a. *IF(CONDITION STATEMENT, IMAGE("/resource/1699269579000/GraphicsPackNew/silk/16/silk/cross.png","", 20,20), IMAGE("/resource/1699269579000/GraphicsPackNew/silk/16/silk/tick.png","", 20,20))*
2. Add this formula field as a column:
   a. *{"label":"LABEL", "fieldName":"record.IMAGE_FORMULA_FIELD", "value":"IMAGE_FORMULA_FIELD", "type":"ra_formulaImage", "sortable":false, "initialWidth": 200, "editable":false, "typeAttributes": {"rowId": { "fieldName": "record.Id"}, "field":"IMAGE_FORMULA_FIELD"}}*
3. Include the formula field *IMAGE_FORMULA_FIELD* in the SOQL query on the Grid record.

You can follow the steps [here](#)

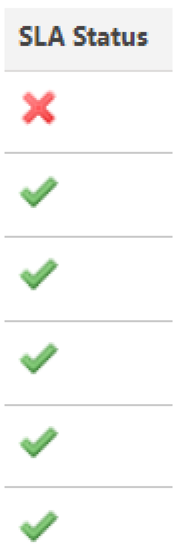
Example 1

1. Create formula field '*SLA_Status__c*' from the images [here](#):

a. *IF(Planned_Date__c <NOW(), IMAGE("/resource/1699269579000/GraphicsPackNew/silk/16/silk/cross.png","", 20,20), IMAGE("/resource/1699269579000/GraphicsPackNew/silk/16/silk/tick.png","", 20,20))*

2. Created the column in the grid record:
    a. *{"label":"SLA Status", "fieldName":"record.SLA_Status__c", "value":"SLA_Status__c", "type":"ra_formulaImage", "sortable":false, "initialWidth": 200, "editable":false, "typeAttributes": {"rowId": { "fieldName": "record.Id"}, "field":"SLA_Status__c"}},*
3. Add the field *SLA_Status__c* into the SOQL query

This gives the SLA status column as below:



4. Loading Custom Images

You can load custom images and use these in formula fields by following the steps [here](). Remember that the File url used below in yellow highlight is for the ContentDocumentVersion.

To create create a dynamic image cell using Salesforce sample images follow the 3 steps below:

1. Upload an image to Salesforce Files and find the ContentVersion record's id using Dev Console / Workbench *SELECT Id FROM ContentVersion WHERE ContentDocumentId = 'ID of File just loaded'*
2. Create a formula field using the referenced file , e.g.
    a. *IF(CONDITION STATEMENT, IMAGE("/resource/1699269579000/GraphicsPackNew/silk/16/silk/cross.png","", 20,20),*

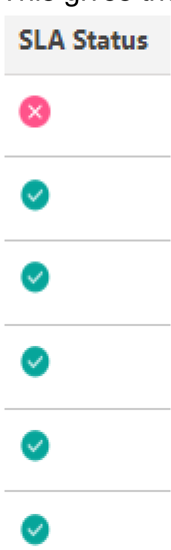
IMAGE("/resource/1699269579000/GraphicsPackNew/silk/16/silk/tick.png","", 20,20))

3. Add this formula field as a column:
    a. *{"label":"LABEL", "fieldName":"record.IMAGE_FORMULA_FIELD", "value":"IMAGE_FORMULA_FIELD", "type":"ra_formulaImage", "sortable":false, "initialWidth": 200, "editable":false, "typeAttributes": {"rowId": { "fieldName": "record.Id"}, "field":"IMAGE_FORMULA_FIELD"}}*
4. Include the formula field *IMAGE_FORMULA_FIELD* in the SOQL query on the Grid record.

Example

1. Create formula field '*SLA_Status__c*' from the images here:
    a. *IF(Planned_Date__c <NOW(), IMAGE("/sfc/servlet.shepherd/version/download/0688e00000CuaWRAAZ","", 16,16),"", 20,20), IMAGE("/sfc/servlet.shepherd/version/download/0688e00000CuaWRAAZ","", 16,16))*
2. Created the column in the grid record:
    a. *{"label":"SLA Status", "fieldName":"record.SLA_Status__c", "value":"SLA_Status__c", "type":"ra_formulaImage", "sortable":false, "initialWidth": 200, "editable":false, "typeAttributes": {"rowId": { "fieldName": "record.Id"}, "field":"SLA_Status__c"}},*
3. Add the field *SLA_Status__c* into the SOQL query

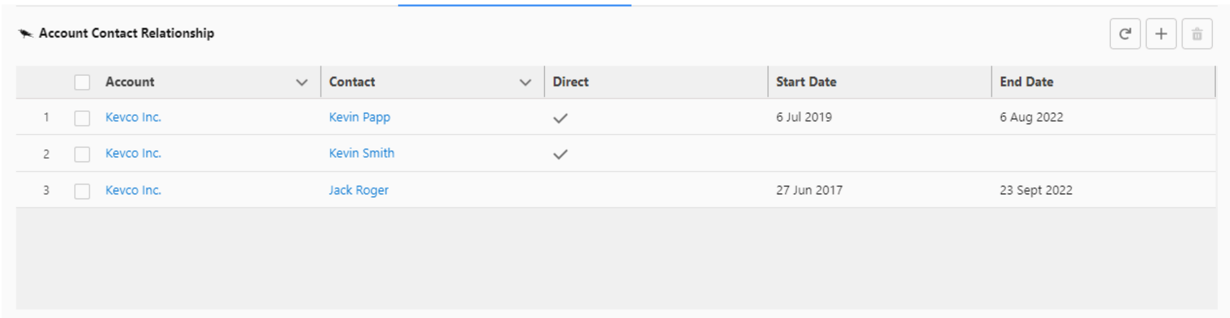
This gives the SLA status column as below:



Working with Junction Objects

When working with junction objects always select the junction object as the Related List object. Then select the Parent Object Field as the parent you want to put this grid on, if you want to put a grid on both parents then this will be 2 separate grids.

For example if you want the Account Contact Relationship grid on the Account as below:


Account Contact Relationship

	Account	Contact	Direct	Start Date	End Date
1	Kevco Inc.	Kevin Papp	✓	6 Jul 2019	6 Aug 2022
2	Kevco Inc.	Kevin Smith	✓		
3	Kevco Inc.	Jack Roger		27 Jun 2017	23 Sept 2022

Then create the Grid on Account and see the Grid detail here.

Working with Record Types

If you wish to use record types on a grid then there are a few ways to do this as described below.

Display Record Type

If you simply need to display a record type on a grid then follow the 2 steps below:

1. Add the below column to the grid which is a read only column to show the record type on the grid
    a. *{"label":"Record Type", "fieldName":"record.RecordTypeId", "value":"RecordTypeId", "type":"ra_lookup", "sortable":true, "initialWidth": 250, "typeAttributes": {"rowId": { "fieldName": "record.Id" }, "object": "RecordType", "icon": "standard:record_lookup", "label": "RecordType", "valueName":{ "fieldName": "record.RecordType.Name" }, "field":"RecordTypeId", "editable":false}}*

2. Add the below to the SOQL query:
    a. *RecordTypeId, RecordType.Name*

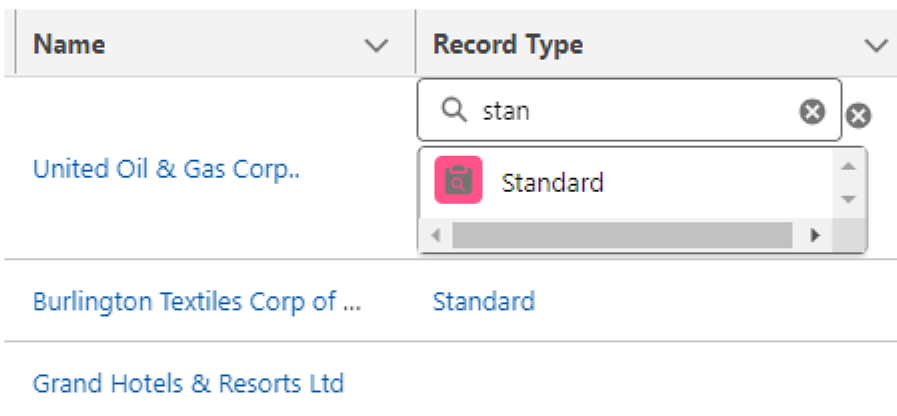
Display and edit Record Type

If you wish to have an editable record type on the grid then follow the 2 steps below:

1. Add the below column to the grid which allows the end user to edit the Record Type:
    a. *{"label":"Record Type", "fieldName":"record.RecordTypeId", "value":"RecordTypeId", "type":"ra_lookup", "sortable":true, "initialWidth": 250, "typeAttributes": {"rowId": { "fieldName": "record.Id" }, "object": "RecordType", "icon": "standard:record_lookup", "label": "RecordType", "valueName":{ "fieldName": "record.RecordType.Name" }, "field":"RecordTypeId", "editable":true, "more":{"whereFilter": "Name IN ('Standard', 'Finance')"}}}*
        i. The text in yellow should be updated with the record types you wish to be allowed to be selected on the grid. If you want all values for that object that place all values in the yellow text.

2. Add the below to the SOQL query:
    a. *RecordTypeId, RecordType.Name*

This will appear as below:



Default Record Types

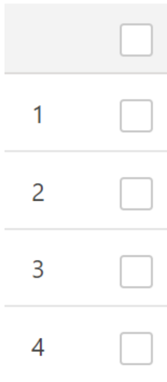
If you wish to auto populate the record type then follow the steps below:

1. Place a grid column on the grid as per 1 of the example above.
2. Populate the 'New Record Row Defaults' field with:
    a. {"field":"RecordType.Name", "type": "value", "value":"Standard"}, {"field":"RecordTypeId", "type": "value", "value":"0128d000000c2gzAAA"}
        i. The text in yellow represents the Record Type and RecordTypeId to default to

Show / Hide Row Number and Checkbox Columns

You can choose to show / hide the numbers and checkboxes on the left of the grid using the fields:

- Show Row Number Column
- Hide Checkbox Column



Please note that if any of the columns are editable then the row numbers can not be hidden, you must ensure each column has *"editable":false* for the number column to be able to be hidden

Visible on Explorer

Checking this box allows users to view the Grid using the RavenApps Explorer tab detailed below. It is advised to only check this box Explorer Grids as Related Lists Grids need a record to reference.

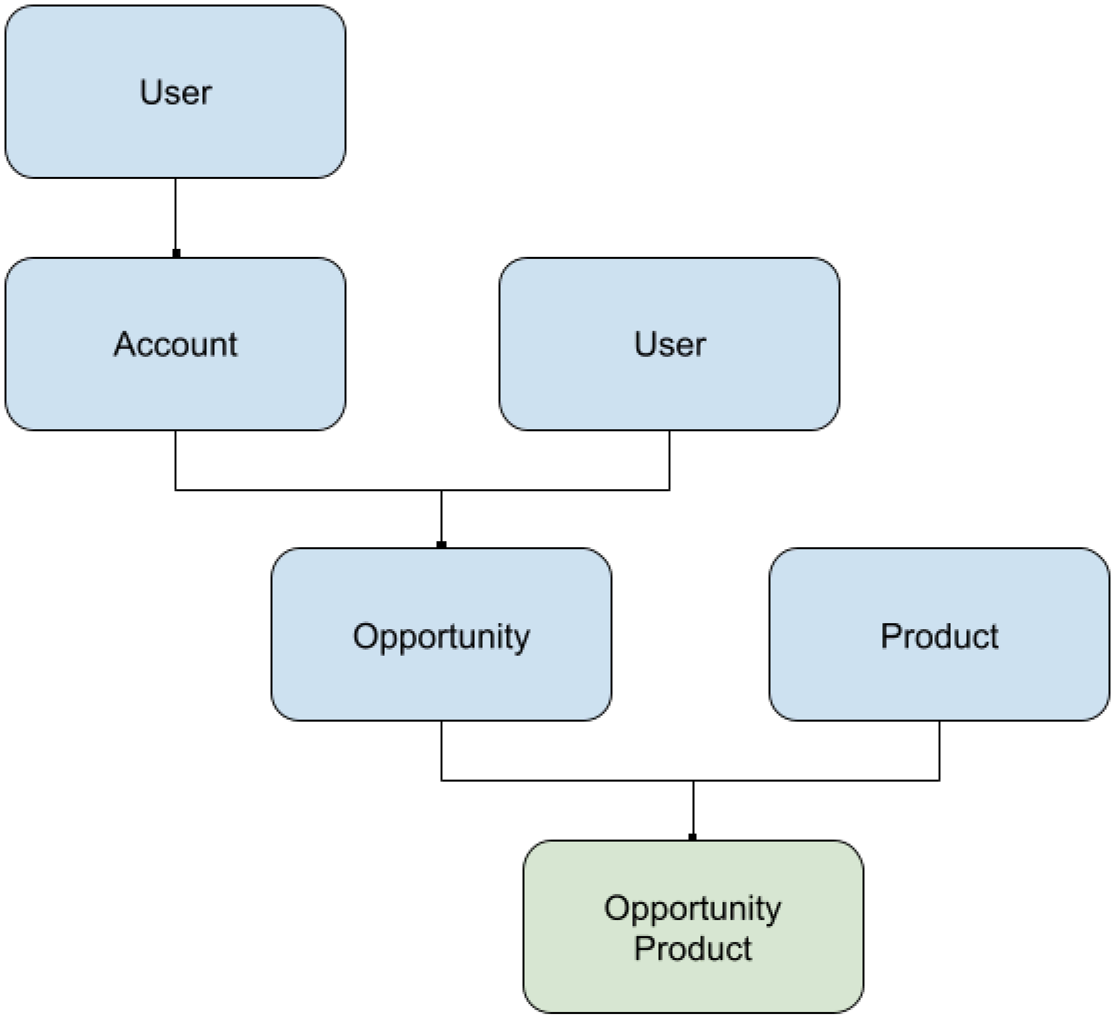
Object Relationships Queries

Grids can handle many types of relationships below we highlight the considerations for different scenarios.

In the below diagram the green object is the basis for the grid (for the 'Object API Name' field and the SOQL query) and the connector indicates a 1 to many relationship using the connector below:

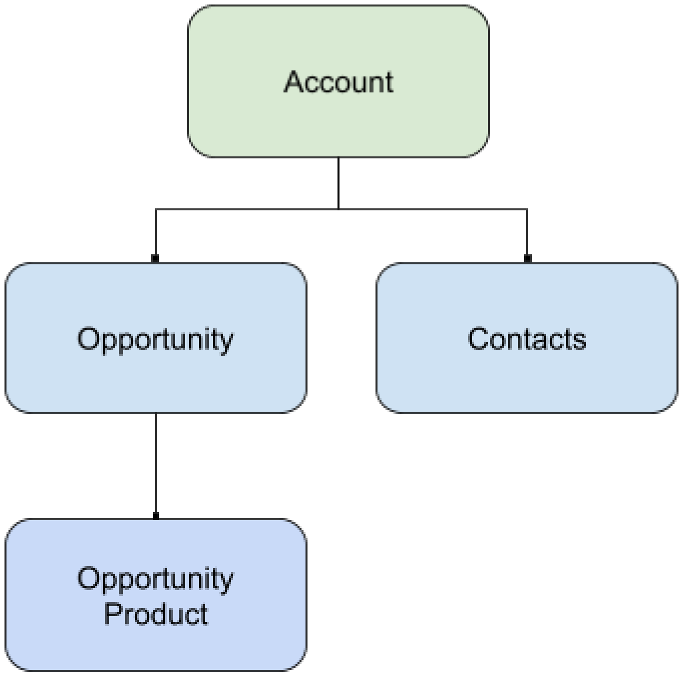
Parent relationship
All fields from parent objects can be pulled into the grid using the standard SOQL child-to-parent relationship queries (see here). This supports up to 5 levels of parent child relationship as per the Salesforce limitation here.

For example if you have a grid on Opportunity Product you can pull in fields from the parent Opportunity, Account, Product and User (owner of these records) into the Grid.



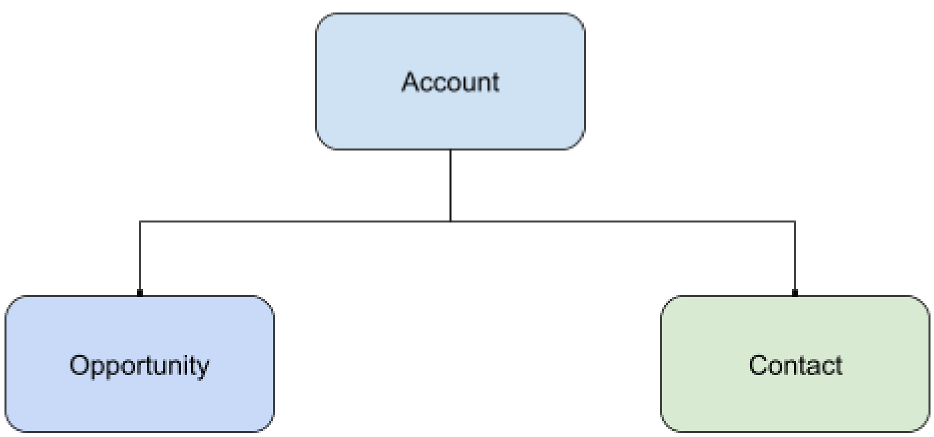
Child relationship
You can go down the object chain by using Related Grids, see details here of how to use these grids.

For example if you have a grid on Account you can can related child grids for Opportunities, Opportunity Product and Contacts.



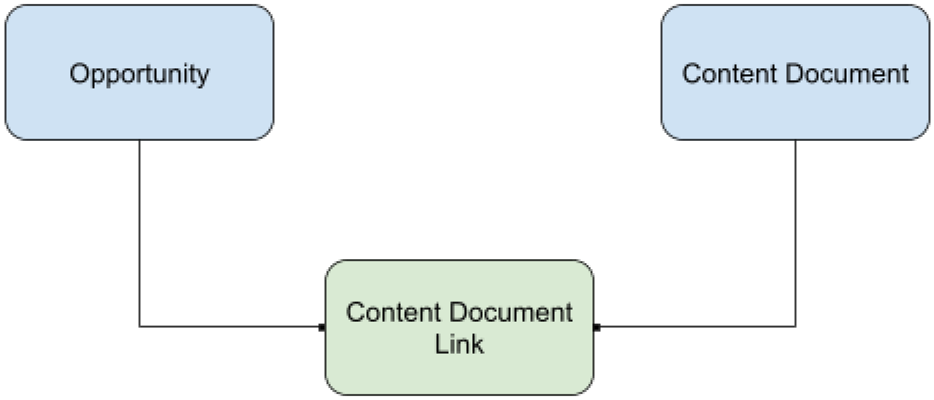
Parent and Parent Child relationships
If you are placing a grid on a record page you can pull in related records of the parent record.

For example you can put a grid on a Opportunity page that shows all Contacts from the parent Account (and add a filter so only show key decision maker from the parent Account).



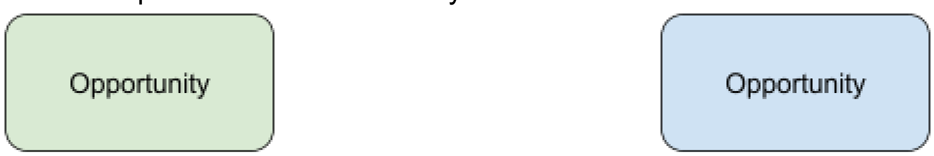
Junction objects
Much like the parent relationship above you can build a grid on a junction object and pull in information from both parents.

For example a grid on Content Document Link surfaces the files that have been stored against records and you can pull in any information from the record (in this case Opportunity) and the file itself (Content Document)



Unrelated Records
Unrelated records can be connected by using variables to connect the records.

For example you can place a grid on the Opportunity page that shows all other Opportunities where the parent account's industry is the same.



SOQL Query

The SOQL query brings in the data that you can run your grids off
[Advanced Queries in Salesforce (Part 1)]
[Advanced Queries in Salesforce (Part 2)]

Salesforce LWC Datatable

RavenGrids leverages Salesforce LWC Datatable component, more details on this component can be found here::
https://developer.salesforce.com/docs/component-library/bundle/lightning-datatable/documentation

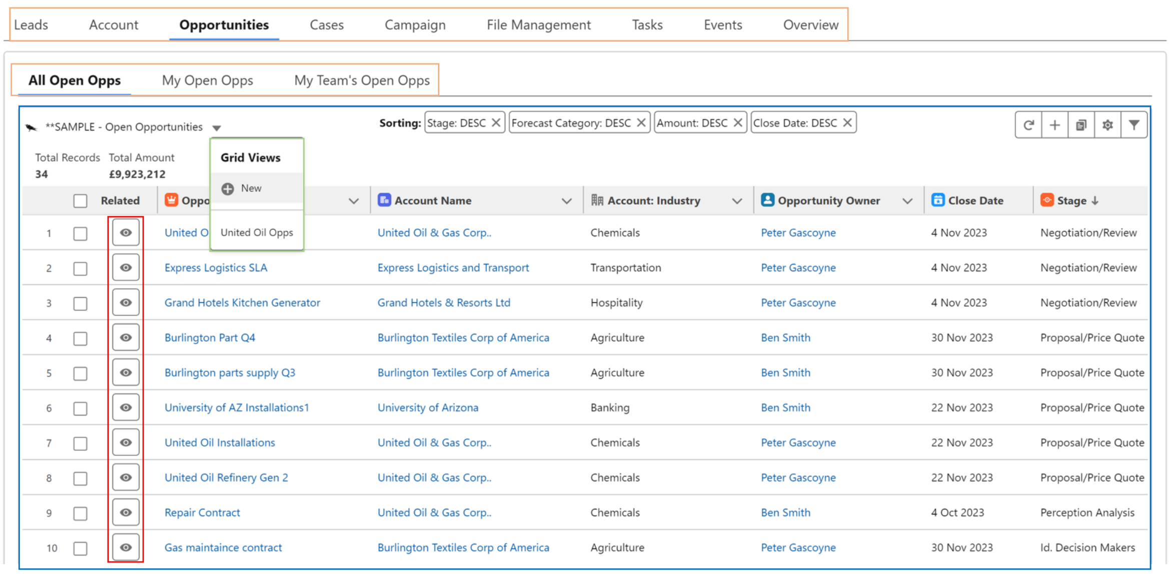
Working with Column Properties

Use the following column properties to customize the behavior and visual aspects of your columns.

PROPERTY	TYPE	DESCRIPTION
actions	object	Appends a dropdown menu of actions to a column. You must pass in a list of label-name pairs.
cellAttributes	object	Provides additional customization, such as appending an icon to the output. For more information, see **Appending an Icon to Column Data**.
editable	boolean	Specifies whether a column supports inline editing. The default is false.
fieldName	string	Required. The name that binds the columns attributes to the associated data. Each columns attribute must correspond to an item in the data array.
fixedWidth	integer	Specifies the width of a column in pixels and makes the column non-resizable.If both `fixedWidth` and `initialWidth` values are provided, `initialWidth` is ignored.
hideDefaultActions	boolean	Specifies whether to hide the default header actions on a column. The default is false. For more information, see **Creating Header Actions**.
iconName	string	The Lightning Design System name of the icon. Names are written in the format standard:opportunity. The icon is appended to the left of the header label.
initialWidth	integer	The width of the column when it's initialized, which must be within the `min-column-width` and max-column-width values, or within 50px and 1000px if they are not provided.
label	string	Required. The text label displayed in the column header.
sortable	boolean	Specifies whether the column can be sorted. The default is false.
type	string	Required. The data type to be used for data formatting. For more information, see **Formatting with Data Types**.
typeAttributes	object	Provides custom formatting with component attributes for the data type. For example, currency-code for the currency type. For more information, see **Formatting with Data Types**.
wrapText	boolean	Specifies whether text in a column is wrapped when the table renders. Wrapped text vertically expands a row to reveal its full content. Use with `wrap-text-max-lines` to display a number of lines before hiding the rest. For more information, see **Text Wrapping and Clipping**.

RavenApps Sample Grids

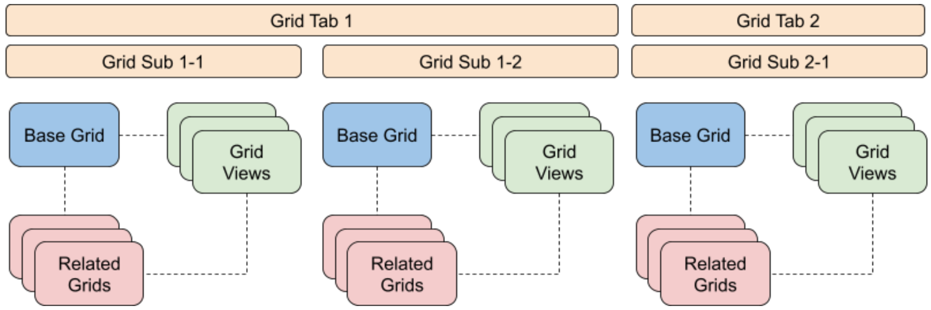
The managed package contains a number of sample grids and these are packaged into a Lightning App Page called Grids CRM. This tab allows you to quickly see the power of grids by clicking through the main base Salesforce objects (Lead, Account, Opportunity, Case etc…). Below we breakdown the different areas of these sample grids:



The 4 different Grid Types in CRM Grids are:

Grid Type	Description
Grid Tabs / Sub Tabs	A tabs contains sub tabs and each sub tab contains 1 base grid
Base Grid	The main grid that appears under a tab / sub tab
Grid Views	Each Base Grid can have many Grid Views that show different fields, filters, sorts etc…
Child Grids	Each Base Grid or Grid View can have many Related Grids that show information on child, grandchild, etc.. records

The below infographic show the relationships between the different grid components:



Grids and CICD / Source Control

Grids uses a Unique Identifier field (RavenApps__UID__c) that for new grids created in an environment will:

- Allow this value to be defined (e.g. by CICD scratch org create), otherwise:
- Will default to be the records Salesforce unique id

This allows scratch org creation and deployments between environments to avoid manual steps once these pipelines have been created.

Grid Field	Example where record created in environment	Example where record created from CICD	Can be updated?	Used for:
RavenApps__UID__c	a0W4K000006mGYZUA2	a0W4K000006nBKJSY2 OR *Opportunity_Hygiene_Main_Grid*	Yes	The field referenced in the Grids Component. When you deploy from Sandbox to Production this value is maintained in the Grid and on the Lightning Page reference to the Grid. You can want to update this value to a unique name that is easier to understand, e.g. Opportunity_Hygiene_Main_Grid.
RavenApps__Grid_id__c	a0W4K000006mGYZUA2	a0W4K000006nBKJSY2 OR *Opportunity_Hygiene_Main_Grid*	No	Formula field on Grids page to show user the id to use or the Grid Component while hiding the editable field 'RavenApps__UID__c'.
RavenApps__RavenAppsId__c			No	Is used by RavenApps to manage the sample grids / CRM Grids in your org. This should not be updated.
id	a0W4K000006mGYZUA2	a0W4K000006mGYZUA	No	The SF Grid record unique id

		2		

The above UID__c field was introduced from version 4.91 onwards - to upgrade see steps here.

Using Scratch Orgs

You can create scratch orgs with Grids by following the below steps:

1. Include the Grids package in your scratch org definition file using the latest version 5.26 which has package id of 04tP10000000Y9IIAE
2. Then in your scratch org create your grid
3. On relevant lightning pages add the Grids LWC component and reference the grid id from point 2
4. Pull your metadata changes into your repository
5. To set up automatic Grid data record creation you can follow the steps here

Optional at point 2 you can update the Grid record's Unique Identifier ('RavenApps__UID__c') to be a defined name for the grid rather than a Salesforce id so it is easier to track in the future (e.g. you could update name to Opportunity_Management).

Performance Optimisation

Grids have very good performance but if you are experiencing the page taking time to load then here are some area to review:

1. If you have an object with a large amount of records (100,000s) of records and you only need the Grid to ever show you a subset of this data then ensure you put a WHERE clause in the Query.
    a. E.g. if you only want to see Contacts of record type Finance then add to the end of the Query *WHERE recordtype.name = ‘Finance’*
2. If your Grid is returning back long text fields and / or rich text fields that have a lot of data this can cause the grid to take a little longer to load. In this case ensure that the columns are needed and limit the query (as per point 1 above) appropriately.
3. Having a large number of formula fields or roll up summary fields may affect performance as these have to be calculated as you view the grid.
4. Positive filters are more efficient than negative filters (see here for more details)
    a. E.g. this is more performant: *WHERE Rating IN ('Hot', 'Warm')*
    b. Than this: *WHERE Rating != 'Cold'*

Refactor

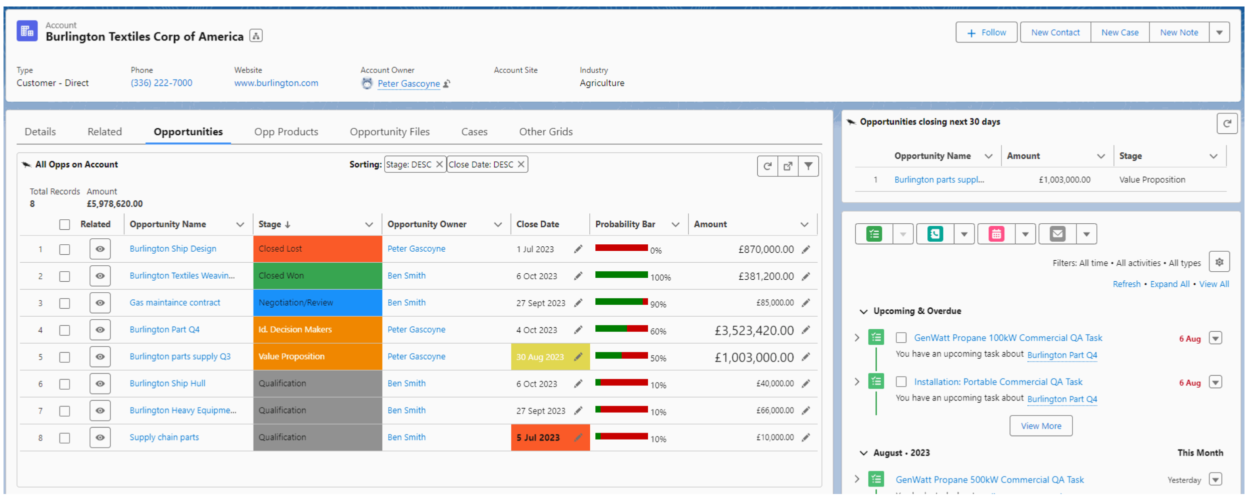
zExamples

[See CRM Grids tab and their Grid Record](#)

This section shows you some example grids and provides the [data records](#) to recreate these grids in your orgs.

Opportunities on Accounts

This grid is placed on the Account record and provides visual tips to help uses see key information including that needs to be updated, e.g. Close Date in the past is highlighted red and close date in next 2 weeks is highlighted yellow. It also contains 2 related grids which allow users to see the Products being sold and tasks on the Opportunity:



The data for the above grids can be found [here](#)

This grid contains the following custom fields on the Opportunity object:

- Lead_Source_Color__c
    - Formula (Text)
    - *IF(ISPICKVAL(LeadSource, "Web"), "slds-icon-standard-empty",*
      IF(ISPICKVAL(LeadSource, "Phone Inquiry"), "slds-icon-custom-custom110" ,
      IF(ISPICKVAL(LeadSource, "Purchased List"), "slds-icon-standard-related-list slds-text-color_default",
      IF(ISPICKVAL(LeadSource, "Other"), "slds-icon-standard-group-loading slds-text-color_inverse",
      IF(ISPICKVAL(LeadSource, "Partner Referral"), "slds-icon-custom-custom46 slds-text-color_default", "")
)
      )))
- Stage_Color__c

    - Formula (Text)
    - *IF(OR(ISPICKVAL(StageName, "Prospecting"),ISPICKVAL(StageName, "Qualification"),ISPICKVAL(StageName, "Needs Analysis")),
      "slds-icon-standard-empty",
      IF(OR(ISPICKVAL(StageName, "Value Proposition"),ISPICKVAL(StageName, "Id. Decision Makers"),ISPICKVAL(StageName, "Perception Analysis")),
      "slds-icon-custom-custom110",
      IF(OR(ISPICKVAL(StageName, "Proposal/Price Quote"),ISPICKVAL(StageName, "Negotiation/Review")),
      "slds-icon-standard-goals",
      IF(ISPICKVAL(StageName, "Closed Won"), "slds-icon-standard-voice-call",
      IF(ISPICKVAL(StageName, "Closed Lost"), "slds-icon-standard-quip", "")
      ))))*
- Amount_Text_Formula__c
    - Formula (Text)
    - If(Amount <100000,"slds-text-body_small",
      IF(Amount<1000000,"slds-text-heading_small",
      "slds-text-heading_medium"
      ))
- Close_Date_in_Past_Color__c
    - Formula (Text)
    - *IF(AND(CloseDate <TODAY(),IsClosed =FALSE), "slds-icon-standard-quip slds-text-color_default slds-text-title_bold",
      IF(AND(CloseDate <(TODAY()+14),IsClosed =FALSE),
      "slds-icon-custom-custom4",
      ""))*
- Probability_Bar__c
    - Formula (Text)
    - *IMAGE("/img/samples/color_green.gif","green",10,80*Probability) &
      IMAGE("/img/samples/color_red.gif","red",10, 80-80*Probability) &
      " " & TEXT(100*Probability) & "%"*

This grid contains the following custom fields on the Activity object for the related Grid 'Open Opportunity Tasks':

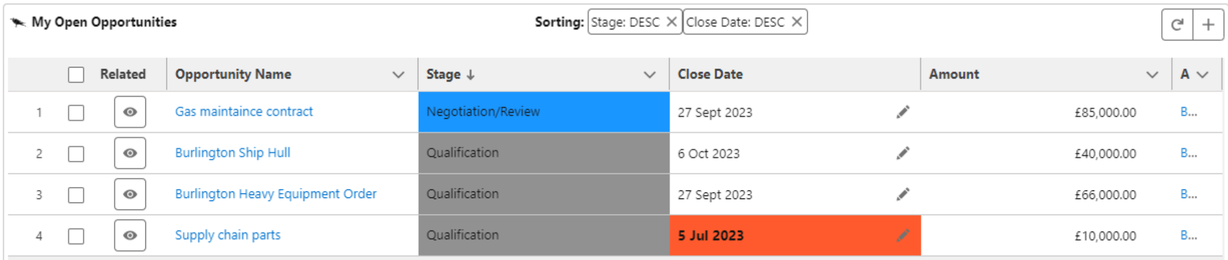
- Due_Date_Color__c
    - Formula (Text)
    - *IF(AND(ActivityDate < Today(),IsClosed = false),
      "slds-icon-standard-quip", "")*
- Priority_Background_Colour__c
    - Formula (Text)
    - *IF(ISPICKVAL(Priority, "High"), "slds-icon-custom-custom112 slds-text-title_bold",
      IF(ISPICKVAL(Priority, "Normal"), "slds-icon-standard-group-loading",
      IF(ISPICKVAL(Priority, "Low"), "", "")
)*

)

Please note when working with Opportunity Products if you update the Quantity then Salesforce keeps the Total Price (TotalPrice) constant and flexes the Sale Price (UnitPrice). This is their expected behavior as described here, if you want to have the Sale Price remain fixed and the Total price flex when the Quantity is edited then you can do this by including in a before save flow an update to Sale Price to keep it constant: PRIORVALUE([OpportunityLineItem].UnitPrice)

My Open Opportunities on Accounts

This grid is placed on the Account record and shows the user only their own Opportunities



The data for the above grids can be found here

This grid contains the following custom fields as mentioned in example 1 on the Opportunity object:

- Stage_Color__c
- Close_Date_in_Past_Color__c

This grid contains the following custom fields as mentioned in example 1 on the Activity object:

- Due_Date_Color__c
- Priority_Background_Colour__c

Child Accounts

This grid is placed on the Account page and shows any child accounts it has.

Query:

SELECT Id,Name,ParentId FROM Account WHERE Parentid = :sv1

Columns:

{"label":"Name","fieldName":"record.Name","value":"Name","type":"text","sortable":true,"editable":true}

Variables:

{"type":"record", "field":"id", "variable":"sv1"}

Contacts On Same Ultimate Parent Account

This Grid is placed on the Account page and displays all contacts for the account hierarchy of the account you are viewing. It works by using variables sv1 and sv2. 'sv1' is the id of the account you are on. 'sv2' is the id of the parent account. The SOQL works as below:

1. Accountid = 'sv1', i.e. all Contacts for the account you are currently viewing.
2. Accountid = 'sv2', i.e. all Contacts from the parent account to the account you are currently viewing.
3. account.parentId = 'sv1', i.e. all Contacts for the child of the account you are currently viewing.
4. account.Parent.ParentId =: sv1 i.e. all Contacts for the grandchild of the account you are currently viewing.
5. account.Parent.ParentId =: sv2. i.e. all Contacts for the grandchild of the account you are currently viewing.
6. account.ParentId =: sv2 i.e. all Contacts for the child of my grandfather's account, this the Contacts of sibling accounts

Query:

SELECT Id, Name, FirstName, LastName, Title, Account.Name, Email, Phone, accountid, account.ParentId, account.parent.name, LeadSource FROM Contact WHERE (Accountid =: sv1 OR Accountid =: sv2 OR account.ParentId =: sv1 OR account.Parent.ParentId =: sv1 OR (account.ParentId =: sv2 AND account.ParentId != null))

Columns:

{"label":"First Name","fieldName":"record.FirstName","value":"FirstName","type":"text","sortable":true,"editable":true, "initialWidth":130},

{"label":"Last Name","fieldName":"record.LastName","value":"LastName","type":"text","sortable":true,"editable":true, "initialWidth":130},

{"label":"Account","fieldName":"record.AccountId","value":"Account","type":"ra_lookup","sortable":true,"editable":true, "typeAttributes": { "rowId": { "fieldName": "record.Id" }, "object": "Account", "icon": "standard:account", "label": "Account", "valueName":{ "fieldName": "record.Account.Name" }, "field":"AccountId" }, "initialWidth":250},

lookup

{"label":"Parent Account", "fieldName":"record.Account.ParentId", "value":"Account.ParentId", "type":"ra_lookup", "sortable":true, "initialWidth":250, "typeAttributes": {"rowId": { "fieldName": "record.Id" }, "object": "Account", "icon": "standard:record_lookup", "label": "Account Name", "subTitleField":"Industry",

"valueName":{ "fieldName": "record.Account.Parent.Name" }, "field":"ParentId", "editable":false}},

{"label":"Email","fieldName":"record.Email","value":"Email","type":"text","editable":true,"sortable":true, "initialWidth":250},

{"label":"Phone","fieldName":"record.Phone","value":"Phone","type":"text","editable":true,"sortable":true, "initialWidth":150},

{"label":"Title","fieldName":"record.Title","value":"Title","type":"text","editable":true, "sortable":true, "initialWidth":150},

{"label":"Lead Source","fieldName":"record.LeadSource","value":"LeadSource","type":"ra_picklist","typeAttributes": {"rowId": { "fieldName": "record.Id" },"objectDotField": "Contact.LeadSource"},"editable":true,"sortable":true, "initialWidth":150}

<u>Sorts:</u>

{"order":1, "label":"Parent Account: DESC", "field":"Account.Parent.Name", "direction": "desc"},
{"order":2, "label":"Account: ASC", "field":"Account.Name", "direction": "asc"},
{"order":3, "label":"Last Name: ASC", "field":"LastName", "direction": "asc"}

<u>Filters:</u>

{"order":1,"label":"Name","field":"Name","value":"","function":"contains","type":"text"},
{"order":2,"label":"Account","field":"Account.Name","value":"","function":"contains","type":"text"},
{"order":3,"label":"Parent Account","field":"account.parent.name","value":"","function":"contains","type":"text"},
{"order":4,"label":"Email","field":"Email","value":"","function":"contains","type":"text"},
{"order":5, "label":"Lead Source","field":"LeadSource","value":"","function":"contains","type":"picklist","objectDotField": "Contact.LeadSource"}

<u>Summaries:</u>

{"label": "Unique emails", "field":"Email", "type": "COUNT_DISTINCT", "formatStyle": "number"}

Variables:

{"type":"record", "field":"id", "variable":"sv1"},
{"type":"record", "field":"parentid", "variable":"sv2"},
{"type":"record", "field":"Name", "variable":"sv3"}

New Record Row Defaults:

{"field":"AccountId", "type": "variable", "value":"sv1"},
{"field":"Account.Name", "type": "variable", "value":"sv3"}

High Priority Cases

This grid shows all high priority cases, as it has no variables it should be used in a stand alone page of the Explore page.

Some areas to note:

- Many of the fields here are picklists and use the type 'ra_picklist' and 'typeAttributes' section to produce the dynamic picklist functionality.
- A filter has been included which searches based on the values not being equal to the search term.

Query:

SELECT Owner.name,Id,CreatedDate,IsClosed,IsEscalated,MilestoneStatus,Origin,Priority,Reason,Status,Subject,Type FROM Case

Columns:

{"label":"Subject","fieldName":"record.Subject","value":"Subject","type":"text","sortable":true,"editable":true},
{"label":"Priority","fieldName":"record.Priority","value":"Priority","type":"ra_picklist","sortable":true,"editable":true,"typeAttributes": {"rowId": { "fieldName": "record.Id" },"objectDotField": "Case.Priority"}},
{"label":"Origin","fieldName":"record.Origin","value":"Origin","type":"ra_picklist","sortable":true,"editable":true,"typeAttributes": {"rowId": { "fieldName": "record.Id" },"objectDotField": "Case.Origin"}},
{"label":"Status","fieldName":"record.Status","value":"Status","type":"ra_picklist","sortable":true,"editable":true,"typeAttributes": {"rowId": { "fieldName": "record.Id" },"objectDotField": "Case.Status"}},
{"label":"IsEscalated","fieldName":"record.IsEscalated","value":"IsEscalated","type":"boolean","sortable":true,"editable":true},
{"label":"Owner","fieldName":"record.Owner.name","value":"Owner.name","type":"text","sortable":true,"editable":true}

Filters:

{"order":1, "label":"Priority", "field":"Priority", "value":"High", "Function":"contains", "type":"text"},
{"order":2, "label":"Origin not equal to", "field":"Origin", "value":"", "Function":"!=", "type":"text"}

Opportunities on same Ultimate Parent Account

This Grid is placed on the Account page and allows you to see all Opportunities for the account you are viewing and for its children and its parent, this can be modified to show more levels as needed.

It works by using variables sv1 and sv2.:
'sv1' is the id of the account you are on. 'sv2' is the id of the parent account to the account record you are on.
The SOQL query finds all Opportunities where:
1. Accountid = 'sv1', i.e. all Opportunities for the account you are currently viewing.
2. Accountid = 'sv2', i.e. all Opportunities from the parent account to the account you are currently viewing
3. account.parentId = 'sv1', i.e. all Opportunities for the child of the account you are currently viewing

Query:

SELECT Id, Name, StageName, Closedate, Account.Name, accountid, account.Parentid FROM Opportunity WHERE Accountid =: sv1 OR Accountid =: sv2 OR account.ParentId =: sv1

Columns:

{"label":"Name","fieldName":"record.Name","value":"Name","type":"text","sortable":true,"editable":true},
{"label":"Stage","fieldName":"record.StageName","value":"StageName","type":"text","sortable":true,"editable":true},
{"label":"Account","fieldName":"record.Account.Name","value":"Account","type":"text","sortable":true}

Filters:

{ "order":1, "label":"Stage", "field":"StageName", "value":"", "Function":"contains", "type":"text"},
{ "order":2, "label":"Name", "field":"Name", "value":"", "Function":"contains", "type":"text"},
{ "order":3, "label":"Account", "field":"Account.Name", "value":"", "Function":"contains", "type":"Text"},
{ "order":4, "label":"CloseDate", "field":"CloseDate", "value":"", "Function":"contains", "type":"date"}

Summaries:

{"label": "Amount", "field": "Amount", "type":"SUM", "formatstyle": "currency"}

Variables:

{"type":"record", "field":"id", "variable":"sv1"},
{"type":"record", "field":"parentid", "variable":"sv2"}

Potential Conflict Opportunities

This related list should be placed on an Opportunity page and will show all related Opportunities to the Opportunity you are viewing. In this case the parent Account's Industry is used as the link to other Opportunities but any other commonality could be used such as:

- Opportunities in the last 12 months to this Account
- Opportunities within 1km of this Opportunity (using geospatial fields)
- Opportunities selling similar products (using roll up on the Opportunity of types of product sold)

This can be used to inform users of useful sales information or to highlight potential conflicts for this opportunity.

Query:

SELECT Id, Name, Amount, Account.Industry, StageName, CloseDate FROM Opportunity Where IsPrivate = FALSE

Columns:

{"label":"Name","fieldName":"record.Name","value":"Name","type":"text","sortable":true,"editable":true},
{"label":"Amount","fieldName":"record.Amount","value":"Amount","type":"currency","sortable":true,"editable":true},
{"label":"Account Industry","fieldName":"record.Account.Industry","value":"Account.Industry","type":"text","sortable":true,"editable":true},
{"label":"CloseDate","fieldName":"record.CloseDate","value":"CloseDate","type":"date","sortable":true,"editable":true}

Sorts:

{"order":1, "label":"Amount: DESC", "field":"Amount", "direction": "desc"}

Filters:

{"order":1, "label":"Name", "field":"Name", "value":"", "function":"contains","type":"text"},
{"order":2, "label":"Account Industry", "field":"Account.Industry", "value":"sv1", "valueType":"variable", "function":"=","type":"text"}

Summaries:

{"label": "Total Amount", "field":"Amount", "type": "SUM"},
{"label": "Average Amount", "field":"Amount", "type": "AVG"}

Variables:

{"type":"record", "field":"Account.Industry", "variable":"sv1"}

Finance Grid

This Grid allows users to view and edit multiple quote lines in a grid format, as it has no variables it should be used in a stand alone page of the Explore page. Filters can be added to allow the grid to be filtered for any data set needed - e.g. all quote lines closing this month for a team or sales person.

Query:

SELECT Quote.Opportunity.Name, LineNumber, Product2.Name, ListPrice, UnitPrice, Quantity, Subtotal, Discount, TotalPrice, ServiceDate FROM QuoteLineItem where quote.issyncing = true

Columns:

{"label":"Opportunity","fieldName":"record.Quote.Opportunity.Name","value":"Quote.Opportunity.Name","type":"text","sortable":true,"editable":false, "initialWidth":250},
{"label":"Product","fieldName":"record.Product2.Name","value":"Product2.Name","type":"text","sortable":true,"editable":false, "initialWidth":250},
{"label":"Date*","fieldName":"record.ServiceDate","value":"ServiceDate","type":"date-local","sortable":true,"editable":true},
{"label":"List Price","fieldName":"record.ListPrice","value":"ListPrice","type":"currency","sortable":true,"editable":false},
{"label":"Unit Price*","fieldName":"record.UnitPrice","value":"UnitPrice","type":"currency","sortable":true,"editable":true},
{"label":"Quantity*","fieldName":"record.Quantity","value":"Quantity","type":"number","sortable":true,"editable":true,"initialWidth":120,"cellAttributes": { "alignment": "center" }},
{"label":"Subtotal","fieldName":"record.Subtotal","value":"Subtotal","type":"currency","sortable":true,"editable":false},
{"label":"Discount %*","fieldName":"record.Discount","value":"Discount","type":"number","sortable":true,"editable":true},
{"label":"Total Price","fieldName":"record.TotalPrice","value":"TotalPrice","type":"currency","sortable":true,"editable":false}

Sorts:

{"order":1, "label":"Date*: DESC", "field":"ServiceDate", "direction": "desc"}

Filters:

{ "order":1, "label":"Product", "field":"Product2.Name", "value":"", "Function":"contains", "type":"text" }

Summaries:

{"label": "Total Price", "field":"TotalPrice", "type": "SUM", "formatStyle": "currency"},
{"label": "Total Quantity", "field":"Quantity", "type": "SUM", "formatStyle": "decimal"}

All field type

This is a purely illustrative example of how each field type can be used. In this example custom fields have been created with the name of the field using its own field type all in a custom object called Field Type Test Object

Query:

SELECT RecordTypeid, RecordType.name, Currency__c,Checkbox__c,Date__c, Date_Time__c, Email__c, Geolocation_field__c, Geolocation_field__Latitude__s,Geolocation_field__Longitude__s, Multi__c, Number__c, Percent__c, Phone__c, Picklist__c, Text__c, Text_Encrypted__c, Text_Area__c, Text_Area_Long__c, Text_Area_Rich__c, Time__c, Url__c, Account_look_up__c, Contact_M_D__c FROM Field_Type_Test_Object__c

Columns:

{"label":"Checkbox__c","fieldName":"record.Checkbox__c","value":"Checkbox__c","type":"boolean","sortable":true,"editable":true},
{"label":"Formula__c","fieldName":"record.Formula__c","value":"Formula__c","type":"text","sortable":true,"editable":true},
{"label":"Name","fieldName":"record.Name","value":"Name","type":"text","sortable":true,"editable":true},
{"label":"Roll_Up_Summary__c","fieldName":"record.Roll_Up_Summary__c","value":"Roll_Up_Summary__c","type":"text","sortable":true,"editable":true},
{"label":"Currency__c","fieldName":"record.Currency__c","value":"Currency__c","type":"currency","sortable":true,"editable":true,"cellAttributes": { "alignment": "left" }},
{"label":"Date__c","fieldName":"record.Date__c","value":"Date__c","type":"date","sortable":true,"editable":true},
{"label":"Date__c","fieldName":"record.Date__c","value":"Date__c","type":"datelocal","sortable":true,"editable":true},
{"label":"Time__c","fieldName":"record.Time__c","value":"Time__c","type":"date","sortable":true,"editable":true},
{"label":"Time__c","fieldName":"record.Time__c","value":"Time__c","type":"time","sortable":true,"editable":true},
{"label":"Date_Time__c","fieldName":"record.Date_Time__c","value":"Date_Time__c","type":"date","sortable":true,"editable":true},
{"label":"Multi__c","fieldName":"record.Multi__c","value":"Multi__c","type":"text","sortable":true,"editable":true},
{"label":"Picklist__c",
"fieldName":"record.Picklist__c","value":"Picklist__c","type":"ra_picklist","sortable":true,"editable":true, "typeAttributes": {"rowId": { "fieldName": "record.Id" },"objectDotField": "Account.Type"}},
{"label":"Text__c","fieldName":"record.Text__c","value":"Text__c","type":"text","sortable":true,"editable":true},
{"label":"Text_Area__c","fieldName":"record.Text_Area__c","value":"Text_Area__c","type":"textarea","sortable":true,"editable":true},

{"label":"Text_Area_Long__c","fieldName":"record.Text_Area_Long__c","value":"Text_Area_Long__c","type":"textarea","sortable":true,"editable":true},
{"label":"Text_Area_Rich__c","fieldName":"record.Text_Area_Rich__c","value":"Text_Area_Rich__c","type":"textarearich","sortable":true,"editable":true},
{"label":"Account_look_up__c","fieldName":"record.Account_look_up__c","value":"Account_look_up__c","type":"lookup","sortable":true,"editable":true},
{"label":"Contact_M_D__c","fieldName":"record.Contact_M_D__c","value":"Contact_M_D__c","type":"text","sortable":true,"editable":true},
{"label":"Text_Encrypted__c","fieldName":"record.Text_Encrypted__c","value":"Text_Encrypted__c","type":"text","sortable":true,"editable":true},
{"label":"Geolocation_field__Latitude__s","fieldName":"record.Geolocation_field__Latitude__s","value":"Geolocation_field__Latitude__s","type":"text","sortable":true,"editable":true},
{"label":"Geolocation_field__Longitude__s","fieldName":"record.Geolocation_field__Longitude__s","value":"Geolocation_field__Longitude__s","type":"geolocation","sortable":true,"editable":true},
{"label":"Record Type Name","fieldName":"record.RecordType.name","value":"recordtype.name","type":"text","sortable":true,"editable":false},
{"label":"Record Type ID","fieldName":"record.RecordTypeid","value":"recordtypeid","type":"text","sortable":true,"editable":false}

Sorts:

{"order":1, "label":"Date: DESC", "field":"Date__c", "direction": "desc"}

Filters:

{"order":1, "label":"Picklist__c", "field":"Picklist__c", "value":"", "Function":"contains", "type":"text"}

Summaries:

{"label": "Currency", "field":"Currency__c", "type": "SUM", "formatStyle": "currency"}

Account Contact Relationship Grid on Account

Query:

SELECT Contact.Name, IsDirect, StartDate, EndDate, Roles, Account.Name FROM AccountContactRelation WHERE AccountId =: sv1

Columns:

{"label":"Account","fieldName":"record.AccountId","value":"AccountId","type":"ra_lookup", "typeAttributes": {"rowId": { "fieldName": "record.Id" },"object": "Account","icon":

"standard:account","label": "Account","valueName":{ "fieldName":
"record.Account.Name" },"field":"AccountId"},"editable":true,"sortable":true},
{"label":"Contact","fieldName":"record.ContactId","value":"ContactId","type":"ra_lookup","t
ypeAttributes": {"rowId": { "fieldName": "record.Id" },"object": "Contact","icon":
"standard:contact","label": "Contact","valueName":{ "fieldName": "record.Contact.Name"
},"field":"ContactId"},"editable":true,"sortable":true},
{"label":"Direct","fieldName":"record.IsDirect","value":"IsDirect","type":"boolean","editable
":true,"sortable":true},
{"label":"Start
Date","fieldName":"record.StartDate","value":"StartDate","type":"date-local","editable":tru
e,"sortable":true},
{"label":"End
Date","fieldName":"record.EndDate","value":"EndDate","type":"date-local","editable":true,
"sortable":true}

Sorts:

{"type":"record", "field":"Id", "variable":"sv1"},
{"type":"record", "field":"Name", "variable":"sv2"}

Filters:

{"field":"AccountId", "type": "variable", "value":"sv1"},
{"field":"Account.Name", "type": "variable", "value":"sv2"}