How do I add a table to my custom report?

You need to create a class for your table that extends **Aero\Admin\Reports\ReportTable**and implements the required public columns function. This columns function will be used to return an array containing your tables columns.

```
<?php

namespace Acme\MyModule\Reports\Tables;

use Aero\Admin\Reports\ReportTable;

class OrdersReportTable extends ReportTable
{
   public function columns(): array
   {
       return [

       ];
   }
}
```

Registering a Table with your Report

To register your newly created table with your report you need to ensure the report uses the **Aero\Admin\Reports\Traits\HasTables**trait and that your table class is in the reports protected static $tables array.

```
<?php

namespace Acme\MyModule\Reports;
```

```
use Acme\MyModule\Reports\Tables\OrdersReportTable;
use Aero\Admin\Reports\Report;
use Aero\Admin\Reports\Traits\HasTables;
use Aero\Cart\Models\Order;
use Illuminate\Database\Eloquent\Builder;

class OrdersReport extends Report
{
   use HasTables;

   protected function newQuery(): Builder
   {
       return Order::with([
           'status', 'items', 'currency', 'discounts', 'payments.method',
       ]);
   }

   protected static $tables = [
       OrdersReportTable::class,
   ];
}
```

Adding Columns to your Table

To add columns to your table you need to use the **Aero\Admin\Reports\ReportTableColumn** facade in your tables columns array method.

ReportTableColumn::create Parameters

Type	Description
String	The first parameter is the header of the column.
Closure	The second parameter is a closure that handles the display content for the column. The closure accepts a $row variable and returns a string or view value.
String	Null

```
<?php
```

```
namespace Acme\MyModule\Reports\Tables;

use Aero\Admin\Reports\ReportTable;
use Aero\Admin\Reports\ReportTableColumn;

class OrdersReportTable extends ReportTable
{
   public function columns(): array
   {
       return [
           ReportTableColumn::create('Date', function ($row) {
               if ($row->ordered_at) {
                   return $row->ordered_at->format('D jS M, H:i');
               }

               return view('admin::reports.partials.muted-created-at', compact('row'));
           }),

           ReportTableColumn::create('Order', function ($row) {
               return $row->reference;
           }),
       ];
   }
}
```

Making your Column Searchable

You can use the **setSearchAction**method to define how your column is searched.

Type	Description
Closure	The first parameter is a closure that is called when your column is being used for a search. The closure is passed the query and search term.
String	Null

```
ReportTableColumn::create('Order', function ($row) {
   return $row->reference;
})
->setSearchAction(function ($query, $term) {
```

```
    $term = ltrim($term, '#');

    $query->whereLower('reference', 'like', "%{$term}%");
}, 'Order Reference')
```

Making your Column Not Exportable

To make your column not exportable you can use the **notExportable**method.

```
ReportTableColumn::create('Order', function ($row) {
    return $row->reference;
})
->notExportable()
```

Setting your Columns Export Content

You can use the **setExportContent**method to define how your column should return data when exporting. This is useful for times where instead of returning a view on the Aero Admin you want your column to return raw data.

The **setExportContent**method accepts a closure that is passed the row.

```
ReportTableColumn::create('Date', function ($row) {
    if ($row->ordered_at) {
        return $row->ordered_at->format('D jS M, H:i');
    }

    return view('admin::reports.partials.muted-created-at', compact('row'));
})
->setExportContent(function ($row) {
    if ($row->ordered_at) {
        return $row->ordered_at->format('D jS M, H:i');
    }

    return $row->created_at->format('D jS M, H:i');
})
```

Making your Column Invisible

You can use the invisible method to make your column invisible. Do note that your column will still be visible in an export.

```
ReportTableColumn::create('Order', function ($row) {
   return $row->reference;
})
->invisible()
```

Setting your Columns Position

Columns are ordered by an Integer position value (lowest numbers go first). You can set a column's position using the position method. On the default Aero reports we increment each rows column.

```
ReportTableColumn::create('Order', function ($row) {
   return $row->reference;
})
->position(1)
```

Revision #1
Created 2026-09-02 13:27:25 UTC by Michael Price
Updated 2026-09-02 13:27:25 UTC by Michael Price