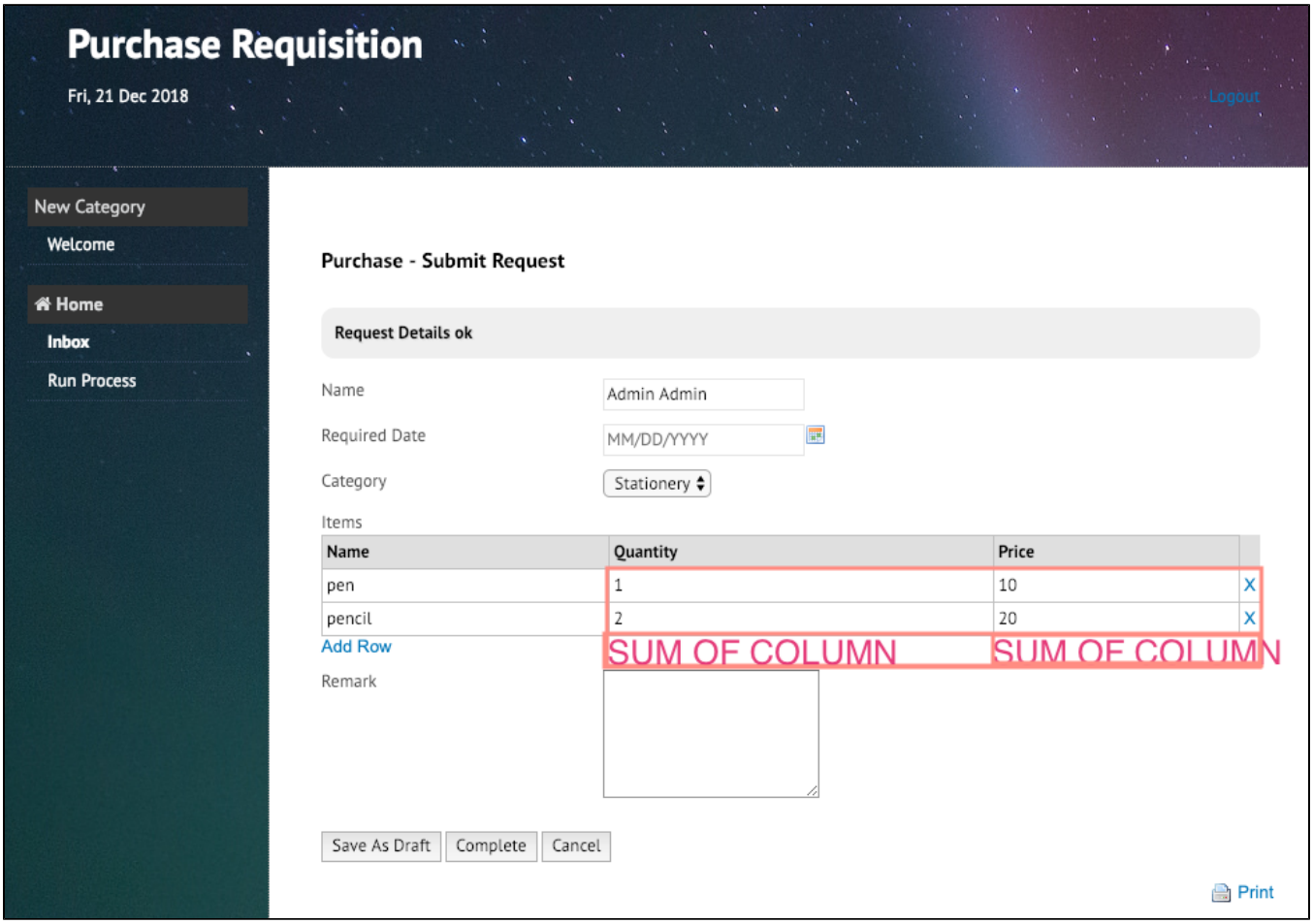
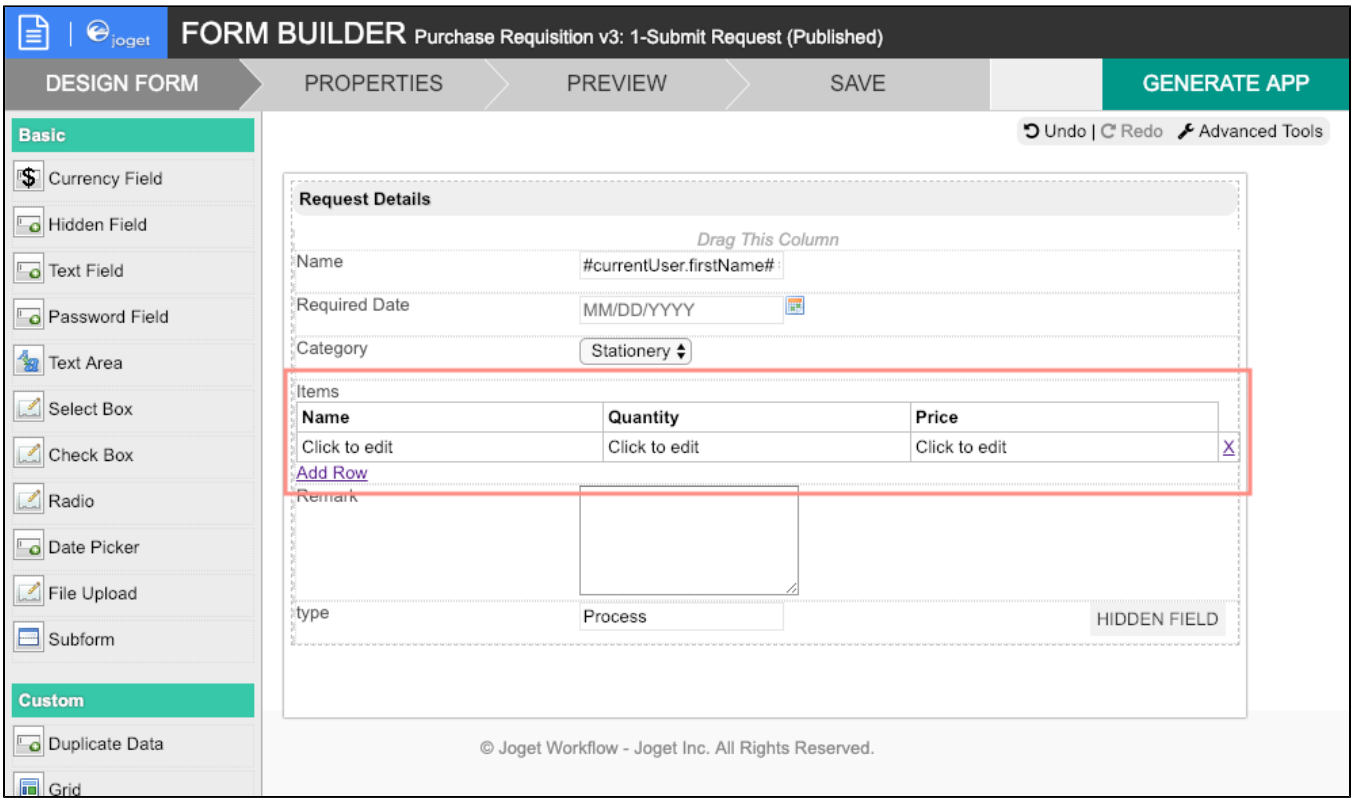
Create Aggregate Final Row in Grid

How can we create a final row that display aggregated values of the rest of the rows? This is especially important when it comes to reporting. One can design the form such that it serves as a report rather than a mean of data entry.

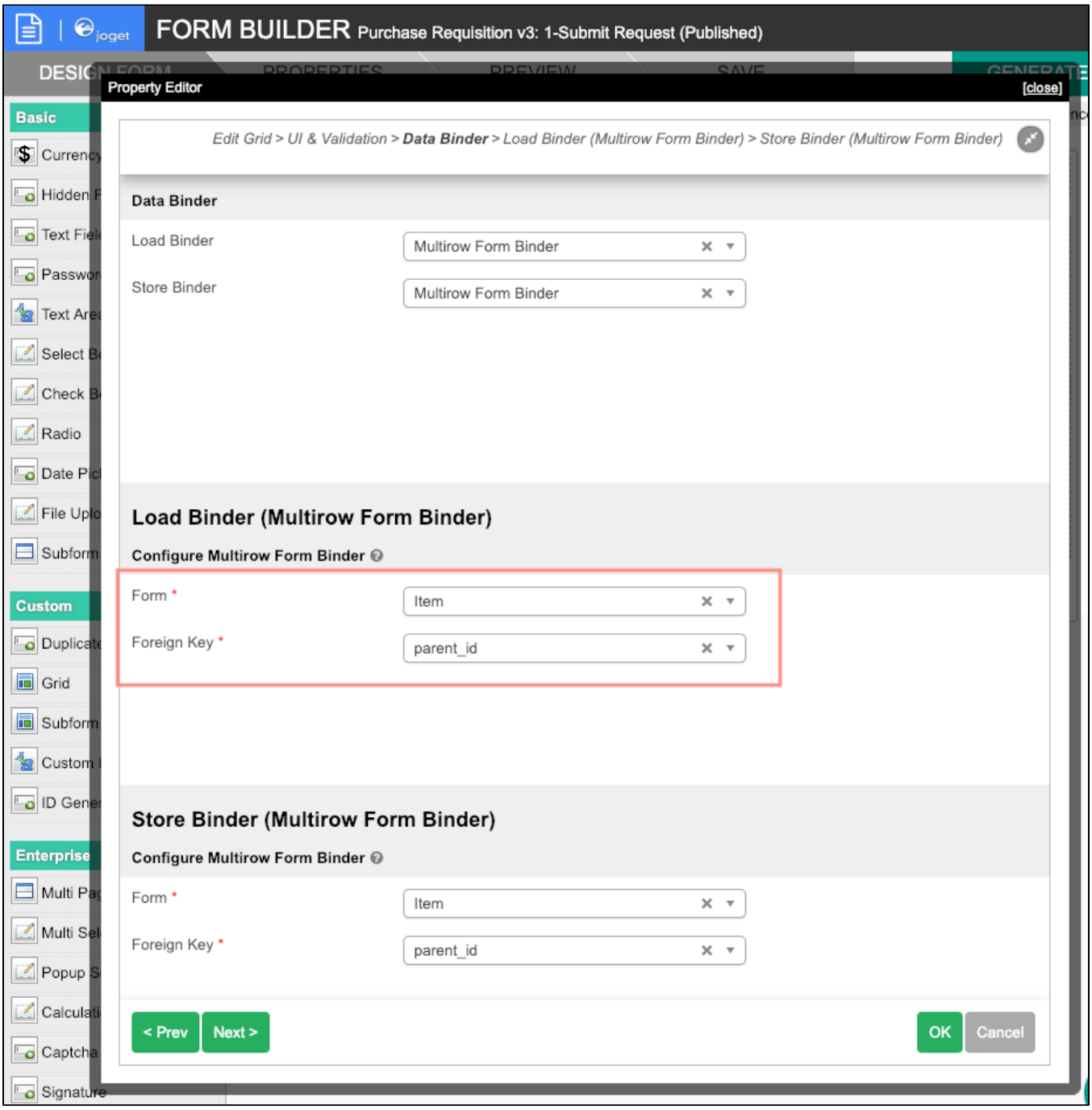
The final row is not a mere record row but serve as a summary of the record rows.



For this to happen, there are many ways to do it but we will focus on the dataset data loaded from its binder. First, we can inspect how data is constructed.



The element in question here is a basic grid, let's click into it.



The grid is using a pair of identical load and store binder (which makes sense most of the time so that data is stored and loaded from the same source). We can opt to use JDBC Binder as the load binder so that we can gain full control on how dataset is returned and constructed.

With the ability to define our own SQL query, we can then compose a query that will always return the last row as an aggregated values row.

```
select c_name, c_quantity, c_price, c_parent_id from app_fd_purchase_items where c_parent_id = ?
union
select "SUM", sum(c_quantity), sum(c_price), c_parent_id from app_fd_purchase_items where c_parent_id = ?
```

And this is a sample result by running them on a command line interface.

```
+--------+------------+---------+--------------------------------+

| c_name | c_quantity | c_price | c_parent_id                    |

+--------+------------+---------+--------------------------------+

| pen    | 1          | 10      | 1177_purchaseRequition_purchase |

| pencil | 2          | 20      | 1177_purchaseRequition_purchase |

| SUM    | 3          | 30      | 1177_purchaseRequition_purchase |

+--------+------------+---------+--------------------------------+
```

However, it does not actually play out well with the JDBC Binder as it is only expecting one parameter in the query. Our union query has 2 parameters.

In order to overcome this, we can create a stored procedure in the database instead.

```
DELIMITER //
CREATE PROCEDURE purchase_items_dataset
(IN recordId CHAR(255))
BEGIN
  select c_name, c_quantity, c_price, c_parent_id from app_fd_purchase_items where c_parent_id = recordId
union
select "SUM", sum(c_quantity), sum(c_price), c_parent_id from app_fd_purchase_items where c_parent_id =
recordId;
END //
DELIMITER ;
```

With the stored procedure to return the appropriate dataset that we need, we will just need to call it from the JDBC Binder.

Load Binder (JDBC Binder)

Configure JDBC Binder

Datasource: Default Datasource

SQL SELECT Query *

```
1 CALL purchase_items_dataset(?);
```

This is the outcome.

Purchase - Submit Request

Request Details

Name: Admin Admin

Required Date: MM/DD/YYYY

Category: Stationery

Items

Name	Quantity	Price	
pen	1	10	X
pencil	2	20	X
SUM	3	30	X

Add Row

Remark

Save As Draft | Complete | Cancel

Print