## ## Introduction

We keep saying it: B2B is hard. Rarely is it a matter of creating a few products, giving them each one price, and opening them up for anyone to buy. There are contracts involved. Often times different buyers see different products at different prices. Sometimes different groups of users within the same buyer organization have varying levels of access to products. Sometimes these variations apply to entire categories of products.

To deal with all this complexity, OrderCloud.io makes heavy use of the concept of *assignments*. Allowing a buyer user to see a product typically means assigning the catalog to the buyer organization, assigning the user to some party (a user group or the entire organization), assigning that party to a category containing the product, and, finally, assigning the party to the product, which must also include a price schedule associated with that party-product relationship.

That’s a fair amount of work, and although it gives you extremely precise control over who sees what, it’s often more control than you really need, which can make catalog management overly burdensome.

This is all about to change. In our next major release, we will be rolling out powerful new features aimed at simplifying catalog management for cases where you don’t need such fine-grained control. Let’s explore these features by use case:

//1 - 6 assumes the product belongs to a catalog that is visible by the parties involved

## ## Scenario 1: "I want control over who sees what, but pricing usually doesn’t vary from buyer to buyer."

For this, we’re introducing <strong>default price schedules</strong>. Just populate `DefaultPriceScheduleID` on the Product model, leave it out of `ProductAssignments`, and those assigned parties will get the default pricing. `ProductAssignment` still has a `PriceScheduleID`, and it takes precedence over the default, but it is now optional. Giving each product a default price schedule is recommended in the majority of cases.

## ## Scenario 2: "I want to allow some users to <em>*see</em>* certain products but not <em>*buy</em>* them."

This is a case where you would <strong>NOT</strong> use a default price schedule. If other visibility requirements are met but neither `Product.DefaultPriceSchedule` nor any `ProductAssignment.PriceScheduleID` related to the user is populated, the product will be visible to the user but not purchase-able.

## ## Scenario 3: "When I assign a catalog to a buyer organization, I just want everybody in that organization to see everything in the catalog."

If your visibility rules are fairly simple, you’ll appreciate this one. `CatalogAssignment` (the relationship between Catalog and Buyer organization) has 2 new properties: `ViewAllCategories` and `ViewAllProducts`. And yup, setting them to `true` does exactly what you think it does. These settings tend to go hand in hand with default price schedules; without those, you’d need to create product assignments anyway just to get pricing in place.

## ## Scenario 4: "I want to control visibility at the category level, but when I assign a category, I want all products and subcategories within it to be visible to the assigned party automatically."

Here we get a little more fine-grained with assignments, but still not all the way down to individual products. You’ll want `CatalogAssignment.ViewAllCategories` and `CatalogAssignment.ViewAllProducts` set to `false`, and set `CategoryAssignment.Visible` and `CategoryAssignment.ViewAllProducts` to `true` as needed. Note that while `CatalogAssignment` can only be applied to an entire buyer organization, `CategoryAssignment` can apply to the entire organization or a user group within it.

## ## Scenario 5: "I want everybody to see everything except a specific category."

`CategoryAssignment.Visible` and `CategoryAssignment.ViewAllProducts` have 3 possible values: `true`, `false`, or `null`. `null` means “inherit this setting from my parent”, i.e. the parent category or, in the case of top-level categories, the catalog. This is a very powerful concept - it means you turn on `ViewAllCategories` and `Vi.ewAllProducts` on the `CatalogAssignment`, and then turn off either or both anywhere down the category tree. Category-level settings, whether true or false, will always override their corresponding catalog-level settings.

## ## Scenario 6: "I want to temporarily hide an entire catalog from all buyers it’s assigned to."

Product and Category have always had an Active bit, independent of assignments, that allows you to effectively hide it from everyone by setting it to `false`. We are now adding an Active bit at the Catalog level as well.

## ## Scenario 7: "I want a search-driven catalog where products do not belong to categories."

We’ve already discussed that a product’s assignment to a buyer party (`ProductAssignment`) is no longer required for visibility. The same holds true for a product’s assignment to a category. As long as `CatalogAssignment.ViewAllProducts` is `true`, the only requirement for a product to become visible via search or direct link is the existence of a `CatalogProductAssignment`. (Note: This assignment is technically required in all product visibility scenarios. If you do not create one before assigning the product to a category, it will be created implicitly. But if it doesn’t exist before creating a `ProductAssignment`, an error will occur. No more “floating” product-party relationships - the Catalog is King!)

## ## Conclusion

We’re confident that most OrderClould.io developers will find at least a subset of these new features useful. Getting to know them now and taking advantage of them whenever possible will relieve you of many catalog maintenance headaches down the road.