# Integration Spec

## Deliverable

* Submit Python code that creates a data bundle following the spec below
* Code must correctly parse the entirety of all data files
* Code must pass code review for accuracy and style
* Code must create an output that looks something like

{"key": {"integration\_id": "A1,CS,0"}, "display\_name": "A-1 Steak Sauce 5 Oz", "name": "A-1 Steak Sauce 5 Oz", "unit\_of\_measure": "CASE", "is\_active": true, "min\_unit\_size": 1.0, "max\_unit\_size": 1.0, "show\_by\_default": true, "unit\_count": 1.0, "base\_unit\_of\_measure": null, "category": "Condiment", "code": "A1", "description": "Brand: A-1 | Pack Size: 24/5 OZ", "integration\_id": "A1,CS,0", "metadata": {"erp\_unit": "CS", "additional\_info": {"breaker\_flag": "0"}}}

{"key": {"integration\_id": "MIRIN60,EA,1"}, "display\_name": "Kikkoman Aji Mirin Cooking Wine 02137", "name": "Kikkoman Aji Mirin Cooking Wine 02137", "unit\_of\_measure": "EACH", "is\_active": true, "min\_unit\_size": 1.0, "max\_unit\_size": 1.0, "show\_by\_default": true, "unit\_count": 1.0, "base\_unit\_of\_measure": null, "category": "Cooking Wine", "code": "MIRIN60", "description": "Brand: Kikkoman | Pack CS: 6/60 OZ | Pack EA: 1/60 OZ", "integration\_id": "MIRIN60,EA,1", "metadata": {"erp\_unit": "EA", "additional\_info": {"breaker\_flag": "1"}}},

{"key": {"integration\_id": "MIRIN60,CS,0"}, "display\_name": "Kikkoman Aji Mirin Cooking Wine 02137", "name": "Kikkoman Aji Mirin Cooking Wine 02137", "unit\_of\_measure": "CASE", "is\_active": true, "min\_unit\_size": 1.0, "max\_unit\_size": 1.0, "show\_by\_default": true, "unit\_count": 1.0, "base\_unit\_of\_measure": null, "category": "Cooking Wine", "code": "MIRIN60", "description": "Brand: Kikkoman | Pack CS: 6/60 OZ | Pack EA: 1/60 OZ", "integration\_id": "MIRIN60,CS,0", "metadata": {"erp\_unit": "CS", "additional\_info": {"breaker\_flag": "0"}}},

{"key": {"integration\_id": "ALBACORE66,EA,1"}, "display\_name": "Chicken Of The Sea Albacore Tuna (Solid White) Dolphin Safe", "name": "Chicken Of The Sea Albacore Tuna (Solid White) Dolphin Safe", "unit\_of\_measure": "EACH", "is\_active": true, "min\_unit\_size": 1.0, "max\_unit\_size": 1.0, "show\_by\_default": true, "unit\_count": 1.0, "base\_unit\_of\_measure": null, "category": "Seafood", "code": "ALBACORE66", "description": "Brand: Festival | Pack CS: 6/66 OZ | Pack EA: 1/66 OZ", "integration\_id": "ALBACORE66,EA,1", "metadata": {"erp\_unit": "EA", "additional\_info": {"breaker\_flag": "1"}}}



## Background

### How Pepper EDI works

* Keys
  + The key is a **lookup** value to identify a unique record in our database. We generate a set of keys from the data set and look those values up in the database
  + Keys require at least one of the available fields but can include as many as all of the fields
    - Keys need to be consistent across a bundle
    - Reaper will attempt to find an entity that matches all of the specified key fields. Not all key fields are required. If multiple entities match, Reaper will fail
* Data maps
  + The data map is what we use to **set** values in the database
* The keys + data maps = a **bundle**
* Not all fields are required to create a working bundle
  + Data maps require certain fields, other fields can be ignored all together, and some have default values if no other values are passed in

## How to read a spec

* **EDI Field** is the list of fields that exist in the DB -> these are static and do not need to be updated/revised in the spec writing process
* **Data Field** is the list of headers that we’ll be using from a set of client export files/tables. No business logic goes in this field

## **Transformation Notes** is where the spec writer will describe how a field needs to be transformed before being ingested into Pepper

## 

## Project spec

Note that all string values should be stripped before being passed to the model.

## Item

What files will we need to read from? If there are multiple files, please include joining logic

* prices.csv
* leadtime.csv
  + leadtime.Item Number matches to prices.Item Number

General entity level logic:

* SKIP rows (ie do not include in bundle) if **any** of the following conditions are met:
  + prices.Description Line 1 **CONTAINS** “Description”
  + prices.Description Line 1 **CONTAINS** “Charge”
  + prices.Description Line 1 **CONTAINS** “Certificate”
  + prices.Description Line 1 **CONTAINS** “Adjustment”
  + prices.Brand Name **IS NULL**

**Key**

| **EDI Field** | **Data Field** | **Transformation Notes** |
| --- | --- | --- |
| integration\_id | Item Number |  |

**Data Map**

| **EDI Field** | **Data Field** | **Transformation Notes** |
| --- | --- | --- |
| display\_name | Description Line 1 | Remove any “�” and “?” characters |
| name | Description Line 1 | Remove any “�” and “?” characters |
| is\_active (default = TRUE) |  | Use Default |
| min\_unit\_size (default = 1.0) |  | Use Default |
| max\_unit\_size (default = 1.0) |  | Use Default |
| show\_by\_default (default = TRUE) | Brand Name | IF prices.Brand Name EQUALS “Allez” and Prices.Item Number CONTAINS any of the following (G,P,W,F,O,U,P,A,N,R) then false  ELSE true |
| unit\_count (default = 1.0) | Priced by Weight,  Item Avg. Weight | IF **Priced by Weight** = “Y” THEN **Item Avg. Weight** ELSE 1 |
| available\_quantity |  | Use Default |
| base\_price\_in\_micros | *<omit>* |  |
| base\_unit\_of\_measure | Priced by Weight | IF **Priced by Weight** = “Y” THEN “POUND” ELSE NULL |
| category | Class Name |  |
| code | Item Number |  |
| daily\_hide\_at | *<omit>* |  |
| daily\_unhide\_at | *<omit>* |  |
| description | Description Line 1, Brand Name,  Pack Size,  Priced by Weight | Concatenate the following 4 values that are non-empty with “ | “ to generate description:   1. IF the last character in **Description Line 1** = “\*” THEN “Call your sales rep to special order” 2. "Brand: "[**Brand**]” 3. “Pack Size:"[**Pack Siz**e]” 4. IF the last character in **Priced by Weight** = “Y” THEN “Average Weight”   E.g. “Call Your Sales Rep To Special Order | Pack Size: 750Ml | Brand: Monin | Average Weight”  Or “ Pack Size: 750Ml | Brand: Monin” |
| integration\_id | Item Number |  |
| lead\_time\_days | prices.Item Number,  prices.On Hand Quantity,  leadtime.Item Number  leadtime.Required Date | IF prices.**On Hand Quantity** > 0 THEN set lead\_time\_days to null   * Determine associated items:   + Item 1: prices.**Item Number**   + Item 2:     - IF prices.**Item Number** ends with “P” OR “S” or “G” THEN **remove** “P” from prices.**Item Number** (e.g. 1000P = 1000) to detemine associated item     - IF prices.**Item Number** does not end with “P” OR “S” or “G THEN **add** “P” from prices.**Item Number** (e.g. 1000 = 1000P) detemine associated item * IF prices.**On Hand Quantity** > 0 for Item 1 **OR** prices.**On Hand Quantity** > 0 for Item 2 THEN lead\_time\_days = NULL for both Item 1 and Item 2 * IF prices.**On Hand Quantity** <= 0 for Item 1 **AND** prices.**On Hand Quantity** <= 0 for Item 2 THEN look if Item 1 or Item 2 present **leadtime.csv:**   + IF Item 1 or Item 2 **IS** present in **leadtime.csv** then take difference between leadtime.**Required Date** and today and then add 1 day (E.g. Required Date = 4/21/23 and today is 4/18/23 then lead time of 4 days)   + IF Item 1 or Item 2 **IS NOT** present in **leadtime.csv** then set lead\_time\_days = 14 for both Item 1 and Item 2 |
| lead\_time\_cutoff\_time | <*omit*> |  |
| photo\_url\_list | *<omit>* |  |
| unit\_of\_measure | Unit of Measure | Use UOM Mapping Below to set Pepper unit\_of\_measure based on Altamira’s Unit of Measure value |
| upc\_list | *<omit>* |  |
| metadata.erp\_unit | Unit of Measure |  |