DailyInventoryTable

**Overview:**

This macro is responsible for extracting the inventory quantities and production dates from each inventory report in the SharePoint folder and creating a formatted table combining all of the data. The inventory reports are saved from Outlook using another macro.

**Subs:**

DailyInventory

This is the main driver for the macro. It starts by looping through the SharePoint folder, assigning the Product Information sheet to the variable “product\_information\_sheet” and adds the other file names to a collection data structure. The headings for the table are then output in the columns A through I. Next, the program loops through the collection of file names and calls the correct sub routine to extract the data and place into the inventory report excel sheet. Once the raw data from the reports is added to the inventory report, the program fills in missing data like ship-by dates, ax numbers, and prod8 codes. The program loops through each row and based on the brewery, knows which information is missing. To fill in the missing data, the program uses VLOOKUP and Index/Match to find the missing data in the Product Information sheet. The program will output “N/A” if the information cannot be found. Then the program will call the sub “DailyInventoryTableDates” to create an excel table that contains the production and ship-by date information. Once the table is created, it is sorted by ax number and then brewery to reduce the amount of computation for the next sub called. After sorting, the “DailyInventoryNoDates” sub is called which creates a new table that contains no date information.

The program logs what is happening in a text file named “logExcelMacro” to help debug the code if something goes wrong. In addition, there are two error handlers. One is for unexpected errors that have not been accounted for and another for when the program tries to open an excel workbook and fails. The second error handler allows the code to continue, whereas the first will end the program.

SaddlecreekInventory

This sub extracts the data from both the Joliet and Modesto inventory reports. The sub takes a Boolean to determine which report it is. The Modesto report lists the dates as 1 year from production, so this date must be altered. There have been two different formats for the Saddlecreek reports. The only difference is some of the columns are shifted over, but the first step of the program is to determine which version the report is. Next, the program loops through Column B of the report to get the item name. Within the same loop, the program counts how many cells are between each item. This will be used next with the production dates. After all the items are added to the item\_array, the program gets the production dates. The program will loop through each row and add production dates for each item to a temporary array using the start and end cells captured in the previous step. Once all the production dates are added for one item, the temporary array is added to a collection and the program continues with the next item. Next, the program gets the production quantities for each item by data. It accomplishes this in the same way as the production dates: using start and end cell information, a temporary array of quantities, and a collection for the temporary arrays. Next, the program combines the production dates and quantities into one data structure. In this process, if there are multiple productions of the same day for an item, the quantities are added. Finally, all of the information is output to a table that will be copied to the master inventory report.

This sub uses arrays that must be resized. A collection or dictionary data structure should have been used for better efficiency, however time did not permit rewriting the code. In addition, the program goes through 3 for loops to get the information. While I have not tried, it reasonable to believe everything could be accomplished in one for loop and would also improve program efficiency.

Lindner

This sub extracts the information from a lindner csv file created by a python script. The program simply loops through each row and extracts ax number, product name, and quantity and then outputs to a table to be copied to the master inventory report. Lindner does not have any production dates.

cityInventory

This sub extracts the information from any of the three city reports. To start, the report loops through each row and stores each unique product into a dictionary data structure as well as the ax number and city’s product id to a collection. Next, the program will loop through the product\_names dictionary and sum the production quantity for each production date. Finally, the program will output the data structures into a table to be copied to the master table.

brewDetroit

This sub will extract the information from the Strohs report. The Strohs report is very simple so this sub simply loops through the cells and adds the product name and units to a data structure and then outputs to a table to be copied to the master table.

Vermont

The vermont report has empty lines between each item listing, and to use a for loop there cannot be a gap in the range so to start this sub deletes the blank rows. Next, the sub loops through each row and adds the product name and prod8 code to collections. Like Saddlecreek, vermont has had slightly altered formats, so the next step is to determine which row the on hand quantities are in. Then the number of units will be added to a collection, and all of the information will be output to a table to be copied to the master.

The vermont report does not contain any production dates.

newHolland

The new Holland report is extracted from outlook into an excel file. This sub loops through each row and adds the product name, production dates, and units to collections and then outputs to a table to be copied to the master

DailyInventoryNoDates

DailyInventoryTableDates