In this post we’ll work with two more complex tasks related with pulling data from Magento 2 and processing it in R.

**Sample tasks 1 – get names and emails of all customers who have placed no orders in given period of time**

It is quite common task – find customers who haven’t placed order in given time to target them in marketing campaign. Let’s see how this task can be done in R.

Assume that we want to get list of customers who placed no orders in May 2013.

# set parameters  
start\_date <- "2013-05-01 00:00:00"  
end\_date <- "2013-05-31 23:59:59"

**Step 1. Get all orders over this period of time.**

We assume that you are already got your Magento 2 REST API authorization token.

endpoint <- "rest/V1/orders"  
myquery <- list("searchCriteria[filter\_groups][0][filters][0][field]"="created\_at",  
"searchCriteria[filter\_groups][0][filters][0][value]"=start\_date,  
"searchCriteria[filter\_groups][0][filters][0][condition\_type]"="gt",  
"searchCriteria[filter\_groups][1][filters][0][field]"="created\_at",  
"searchCriteria[filter\_groups][1][filters][0][value]"=end\_date,  
"searchCriteria[filter\_groups][1][filters][0][condition\_type]"="lt")  
orders <- getm2objects(urlbase = urlbase, endpoint = endpoint, query = myquery, auth=auth)

**Step 2. Get email of customers who made these orders.**

# initialize empty factor and add there ids of customers who placed orders  
customers\_ordered <- character()  
# run loop  
for (i in 1:length(orders[["items"]])){  
customers\_ordered[i] <- orders[["items"]][[i]][["customer\_email"]]  
}  
# get rid of duplicates and combine in one string  
customers\_ordered <- unique(customers\_ordered)

We need customer emails in that format for future search.

**Step 3. Get customers who are not in this list**

We use there Not In search query parameter, condition time ‘nin’ in Magento 2 REST API

#form character/string for query  
customers\_ordered\_req <- paste(customers\_ordered, collapse = ',')  
# get all who not in the list of customers ordered  
# using customer search endpoint  
endpoint <- "rest/V1/customers/search/"  
myquery <- list("searchCriteria[filter\_groups][0][filters][0][field]"="email",  
"searchCriteria[filter\_groups][0][filters][0][value]"=customers\_ordered\_req,  
"searchCriteria[filter\_groups][0][filters][0][condition\_type]"="nin")  
cust\_no <- getm2objects(urlbase = urlbase, endpoint = endpoint, query = myquery, auth=auth)

No we have all customers who placed no orders in given period of time in cust\_no object.

Step 4. Save results for further use

Finally, lets extract emails, first and last names to data frame and save it to csv file that we can load to our marketing campaign software for targeting.

emails\_list <- data.frame()[1:length(cust\_no[["items"]]),]  
for (i in 1:length(cust\_no[["items"]])) {  
emails\_list$email[i] <- cust\_no[["items"]][[i]][["email"]]  
emails\_list$fname[i] <- cust\_no[["items"]][[i]][["firstname"]]  
emails\_list$lname[i] <- cust\_no[["items"]][[i]][["lastname"]]  
}  
# write as CSV file to use in your marketing software  
row.names(emails\_list) <-NULL  
write.csv(emails\_list, file="email-list.csv", row.names = FALSE)

**Sample task 2. Compare sales for products in 2 categories for the period of time**

**Step 1. Define parameters: start and end date, product categories**

#define dates  
start\_date <- "2013-01-01 00:00:00"  
end\_date <- "2013-12-31 23:59:59"  
# define categories, using categories ids  
cat1 <- "11"  
cat2 <- "6"

**Step 2. Search through API**

# set API endpoint - orders  
#searching invoices after specific date  
endpoint <- "rest/V1/orders"  
Now let’s form search query. Using search with filtered data pulled.  
It is important to have request started from items in filtering part of query. See the last parameter in query  
We only count orders with “complete” status.  
myquery <- list("searchCriteria[filter\_groups][0][filters][0][field]"="created\_at",  
"searchCriteria[filter\_groups][0][filters][0][value]"=start\_date,  
"searchCriteria[filter\_groups][0][filters][0][condition\_type]"="gt",  
"searchCriteria[filter\_groups][1][filters][0][field]"="created\_at",  
"searchCriteria[filter\_groups][1][filters][0][value]"=end\_date,  
"searchCriteria[filter\_groups][1][filters][0][condition\_type]"="lt",  
"searchCriteria[filter\_groups][2][filters][0][field]"="status",  
"searchCriteria[filter\_groups][2][filters][0][value]"="complete",  
"searchCriteria[filter\_groups][2][filters][0][condition\_type]"="eq",  
"fields"="items[increment\_id,status,items[sku,name,row\_total]]")  
orders <- getm2objects(urlbase = urlbase, endpoint = endpoint, query = myquery, auth=auth)

**Step 3. Data wrangling orders data**

No we need to do a bit of data wrangling, will use ***dplyr*** library.  
library(dplyr)  
n <- length(orders[["items"]])  
# unpack list to dataframe  
sales <- data.frame()  
for (i in 1:n){  
m <- length(orders[["items"]][[i]][["items"]])  
for (j in 1:m){  
stemp <- data.frame()[1,]  
stemp$sku[1] <- orders[["items"]][[i]][["items"]][[j]][["sku"]]  
stemp$name[1] <- orders[["items"]][[i]][["items"]][[j]][["name"]]  
stemp$amount[1] <- orders[["items"]][[i]][["items"]][[j]][["row\_total"]]  
stemp$order[1] <- orders[["items"]][[i]][["increment\_id"]]  
sales <- rbind(sales, stemp)  
}  
}  
# change sales to numbers  
# filter out zero values (happens due to configurable products references)  
# and count total for sku  
sales$amount <- as.numeric(sales$amount)  
sales <- sales %>% filter(amount!=0) %>% group\_by(sku) %>% summarise(total=amount, n=n())  
We have a data frame of all products sold over period of time.

**Step 4. Matching products with categories**

We need to find product categories of the products sold over given period of time. Note that in Magento 2 each product can be in more than one category.  
We use filtered search with ***getm2productdata***function defined earlier (in part 2 of this series of articles).  
fields <- "sku,extension\_attributes[category\_links[category\_id]]"  
n <- nrow(sales)  
### this loop takes time to run  
for (i in 1:n){  
t <- getm2singleproductinfo(url=urlbase, sku=sales$sku[i], fields=fields, auth)  
cats <- unlist(t[["extension\_attributes"]][["category\_links"]])  
## check if sku sale falls in any of our 2 target category and add new column with logical flag  
sales$cat1[i] <- cat1 %in% cats  
sales$cat2[i] <- cat2 %in% cats  
}

Now we filter out the results to include the categories we are interested in and calculate total for each.

# calculate total  
sales\_cat1 <- sales %>% filter (cat1)  
cat1totalsales <- sum (sales\_cat1$total)  
sales\_cat2 <- sales %>% filter (cat2)  
cat2totalsales <- sum (sales\_cat2$total)

The numbers we need will be in *cat1totalsales* and *cat2totalsales* variables.

That is it for this blog post. As you see it is possible to use Magento 2 API combined with R provides quite a handy tool set for eCommerce data analysis.