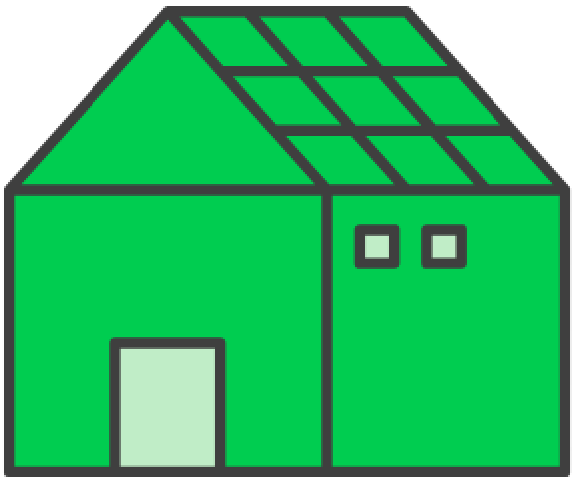


Documentation

Carleon case – Dashboard

Group 4

Anna Chernova, Izabella Bogdanova, Yazan Fattal, Gijs Wijngaards, Sjoerd Heijmann, Tan Quang Nguyen

17/01/2022

Table of Contents

1. Overview

The dashboard's main goal is to let Carleon gain a better understanding of their present financial status by offering some crucial data.

The dashboard's underlying concepts and models, as well as the calculation formulas, will be explained in this document.

2. Dashboard

2.1 Import data

The data are retrieved from *SQL Server* (which is already provided) by using data connectivity mode: *DirectQuery* in the *PowerBI.*

Following the import of raw data. *PowerQuery* makes changes to all the data tables, such as renaming, removing, and unpivoting columns, among other things.

Before producing the graphic, we created the relationship between tables in the model tab, to ensure that our data has a connection and that we will be able to work on filtering.

2.2 General

The dashboard is split into 3 pages which are: *Unpaid invoice progression*, *Debtor division* and *DSO*

There are two types of graphs that can be used to examine how data changes over time: *line charts* and *bar charts*.

Two slicers are included on each page to assist users in filtering out a specific amount of time if necessary.

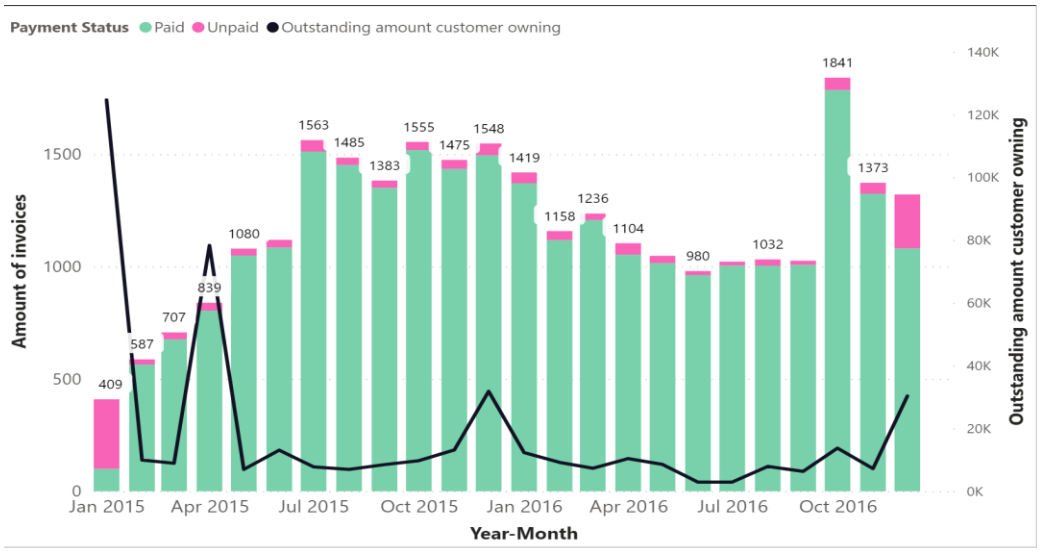
2.3 Scope

The dashboard's purpose is to display our KPIs, which are displayed in the charts below. We managed to emphasize the key performance indicators we pulled from the data that will be essential for the customer on the charts.

We used a variety of techniques and tools to create them, like Power Bi, which is the major one where the dashboard is displayed, and SQL Server, which is where the data is taken. However, to finish our job, we will contribute to RStudio, where validation will be included to ensure that our calculations are proper and may be used in result.

2.4 Charts

Model 1 : Amount of invoices and oustanding amount owning by year-month and payment status.



The stacked columns illustrate the number of paid (green part) and unpaid (pink part) invoices – measured in the right y-axis

The line graph shows the total of outstanding money owning – measured in the right y-axis

Calculations:

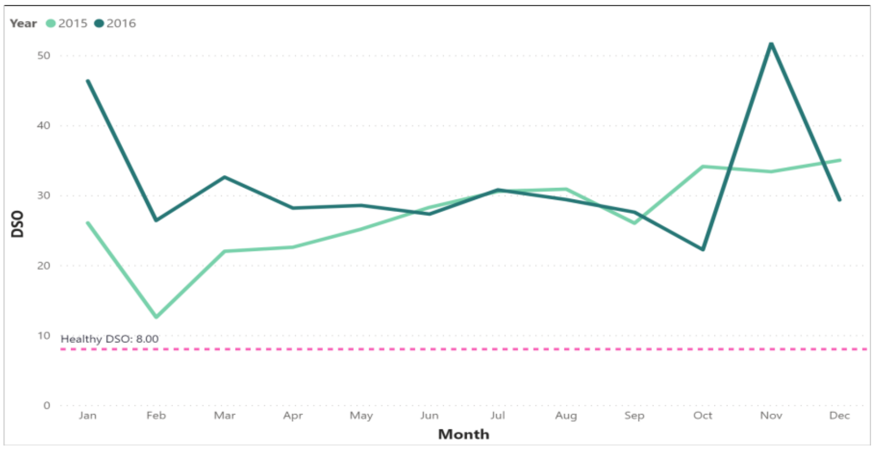
+) An additional column ('Payment Status') is created based on the 'Paid' column in powerbi_balancelist table : If Paid=0→ 'Unpaid', Paid=1→ 'Paid'

+) Using count function for the column 'Payment Status' (COUNT(powerBI_balancelist[Payment Status])) to count the number of invoices with different status

+) The total of outstanding money owning is calculated by summarise all the values greater than 0 in `Outstanding Amount` column in the 'GRV_DebtorOutstanding' table : CALCULATE(SUM(GRV_DebtorOutstanding[Outstanding Amount]),GRV_DebtorOutstanding[Outstanding Amount]>0)

In general, there is not many unpaid invoices in each month-year. It just fluctuates around between 30 and 60 invoices. In some like Jan 2015, Dec 2016, the number of unpaid invoices reached to more than 200. Normally, we will think that the outstanding money will increase if the unpaid invoice increases. It's true, but not entirely, for example the outstanding money still grow up even if the number of unpaid invoices drop from Jun to July in 2017. It is because the outstanding amount is depend on the value of each invoice, not the quantity.

Model 2 : Daily Sales Oustanding



The dark green line represents for the DSO in 2016

The light green line represents for the DSO in 2015

The dashed line is the safe level for DSO (assumption is 8)

Formula: DSO= (Account Receivable/Total Credit Sales) * number of days

Calculations:

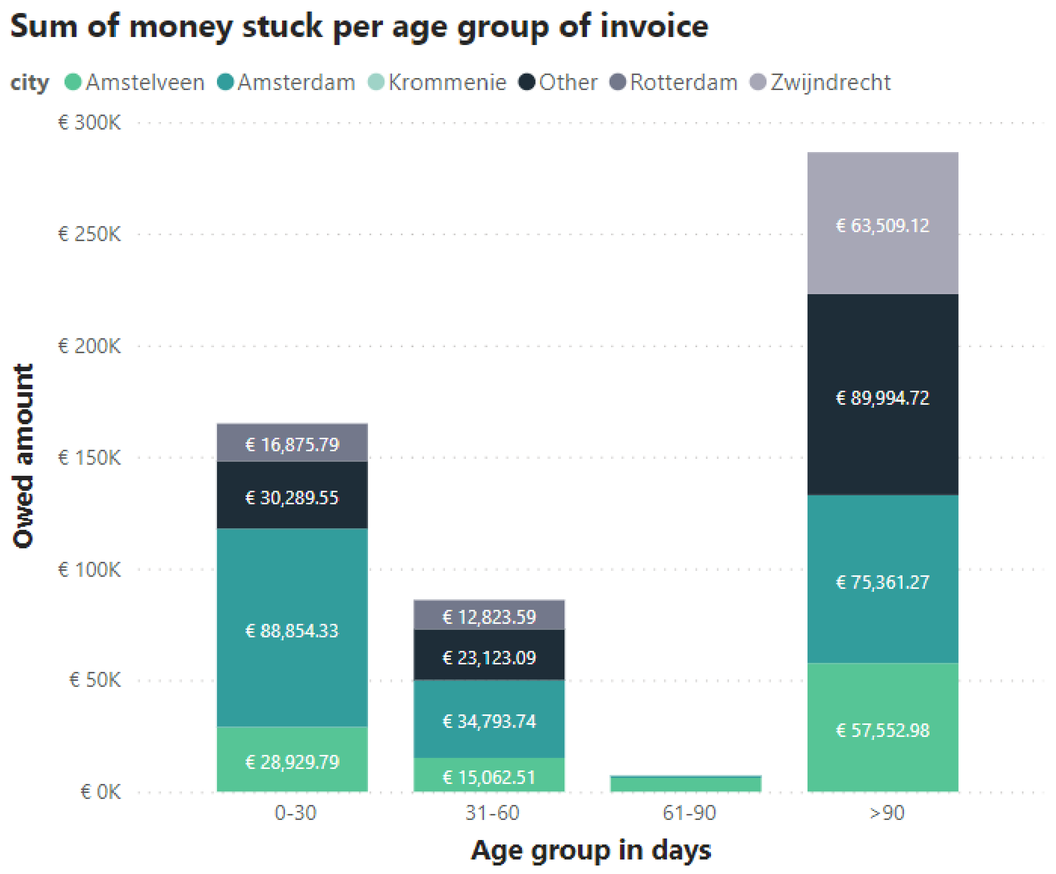
+) Total credit Sales is calculated by summarising the total invoice amount in in powerbi_balancelist table: Credit Sales = SUM(powerbi_balancelist[InvoiceAmount])

+) Account receivable from customers = total credit sales- amount that customer has paid:

[Credit Sales] - CALCULATE(SUM(powerbi_balancelist[ReceiptPaid]), powerbi_balancelist[Paid])

It is important for the company to keep track of the DSO-the average number of days to take the customers paying their bill after receiving invoice. By calculating DSO, the company is able to know how long they can get money from their customer on average, then if it suddenly increases, they can respond quickly before losing some unnecessary money for suppliers because of paying late as customers paying late. In general, the DSO fluctuates around 30 days which is not a big problem, but it should be supposed to decrease. Therefore, the company can get money earlier from customers to pay for suppliers

Model 3 : Sum of money per age group



This graph above shows the sum of outstanding money per age group with top5 city. As we can see, Amsterdam and Amstelveen are cities with the most oustanding money in all 4 groups, especially in the group >90. Most of the owed money also are stuck at this group which is more than 50% of total. It is really worrying as the company cannot use this money to pay for suppliers. It is understandable that bigger cities have more outstanding money, it might be because the company has more property and customer than smaller cities.

The measure "Top 5 and other" calculates the [Sum of money per age], by filtering out only the top 5 cities based on the sum of money measure. If a city is not in the top 5, its value gets added to "other"

Others:

Number of customers calculated by count the distinct code in cicmpy table: DISTINCTCOUNT(cicmpy[cmp_code])

Number debtors calculated by count the distinct code as for the debtors with outstanding amount >0

CALCULATE (DISTINCTCOUNT(GRV_DebtorOutstanding[Debtor code]),GRV_DebtorOutstanding[Outstanding Amount] > 0)

Number of reminders = COUNTROWS(powerbi_reminders)