Jeff1 EDA

Group Members:

Pierce Myers, Aaron Leon, Andrew Vernon, Robert Rehkemper, Ty Mudgistratova

# Setting up environment

library(knitr)  
library(ggplot2)  
library(ggpubr)  
library(dplyr)

##   
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':  
##   
## filter, lag

## The following objects are masked from 'package:base':  
##   
## intersect, setdiff, setequal, union

library(DBI)  
library(factoextra)

## Welcome! Want to learn more? See two factoextra-related books at https://goo.gl/ve3WBa

library(GGally)

## Registered S3 method overwritten by 'GGally':  
## method from   
## +.gg ggplot2

library(ggrepel)  
library(klaR)

## Loading required package: MASS

##   
## Attaching package: 'MASS'

## The following object is masked from 'package:dplyr':  
##   
## select

library(formattable)

##   
## Attaching package: 'formattable'

## The following object is masked from 'package:MASS':  
##   
## area

#Set working directory for code chunks and libraries  
knitr::opts\_knit$set(root.dir = '/cloud/project/PierceSourceData')

## Read Files

partner = read.csv("Partner.csv")  
partnerSol=read.csv("PartnerSolution.csv")  
SolutionAreaMap = read.csv("SolutionAreaMap.csv")  
partnerSolArea = read.csv("PartnerSolutionSolutionArea.csv")  
CustomerWins = read.csv("CustomerWins.csv")  
modwrk = read.csv("Performance-MODWRK.csv")  
bizapp = read.csv("Performance-BIZAPPupdated.csv")  
azure = read.csv("Performance-AZURE.csv")  
activity = read.csv("Activity.csv")  
consumption = read.csv("Consumption.csv")  
partnerSpec = read.csv("PartnerSpecialization2.csv")  
customer = read.csv("Customer.csv")  
individual\_activity = read.csv("IndividualActivity.csv")  
CusKeyParKeyCount = read.csv("CusKeyParKeyCount.csv")

## Partner Data Cleaning:

**Partner table**: Found N/A values in column PartnerSegment. Also duplicates were found and removed, with partner segment and partner key uniquely identifying each row. Categorical variables were converted to factors.

**PartnerSolutions table**: Found N/A values in column OfferStatus. Converted keys and categorical variables to factors.

**PartnerSolutionArea**: Converted both Solution key and SolutionAreaKey to factors. No other data cleaning necessary.

**PartnerSpecialization table**:Converted categorical variables and keys to factors except StartFiscalMonthID was converted to date.

**Performance Files** : There is no missing data in any of the Performance files. One file had 5,639 rows with a NULL but all these values were asked to be changed to 0. There are many outliers but I am keeping them in because they could identify a strong partner. With data structure, even though most data have integers, they are all key-based, meaning all the data, with the exception of Revenue/Active Usage, needed to be categorized as factor.

**Customer**: In Customer, I converted CustomerKey, SalesTerritoryID, and VerticalKey to factors.

**Activity**: I converted ActivityKey to a factor.

**IndividualActivity**:In IndividualActivity, I eliminated a large number of duplicate rows. The number of rows was originally 1571155, trimmed down to 226456, meaning there were 1344699 duplicate rows deleted. In IndividualActivity, I converted PartnerKey, IndividualKey, and ActivityKey to factors.

**CustomerWins**:Converted all categorical variables and keys to factors. Removed duplicates.

**SolutionAreaMap**: Both columns uniquely identify table. There were no duplicates and categorical variables converted to factors.

**Consumption**: Converted CustomerKey, Workload, and PartnerOneKey to factors.

# Data Cleaning Code

#libraries: ggplot2  
#Using is.na function which returns True in case of missing values and False   
#otherwise and which() function to return positions of missing values.   
#No missing values were found however partner segment has N/A values.  
  
which(is.na(partner))

## integer(0)

#Data set does not contain any outliers as each column is categorical or an  
#Identification number.Categorical variables are consistent across table and   
#converted categorical variables and keys to factors.  
  
partner$IsManaged = as.factor(partner$IsManaged)  
partner$IsCosellReadyPartner = as.factor(partner$IsCosellReadyPartner)  
partner$PartnerSegment = as.factor(partner$PartnerSegment)  
partner$PartnerKey = as.factor(partner$PartnerKey)  
partner$ProgramTierKey = as.factor(partner$ProgramTierKey)  
  
#Duplicate rows found in data and removed.  
#Code identifies there is duplicate values in primary key PartnerKey.   
#True if there are no duplicates otherwise False.  
  
dim(partner)[1]==dim(unique(partner))[1]

## [1] FALSE

#Get rid of duplicate rows  
CleanedPartner = partner[!duplicated(partner[,c(1,4)]),]  
  
#Acknowledge 10 partner keys are duplicated but have different PartnerSegment   
#So PartnerKey and PartnerSegment can serve as composite key to uniquely identify   
#each row.  
  
CleanedPartner[duplicated(CleanedPartner$PartnerKey),]

## PartnerKey IsManaged IsCosellReadyPartner PartnerSegment ProgramTierKey  
## 5618 1148041 Yes Yes Services 92315523  
## 7855 1090284 Yes Yes ISV 92315523  
## 27365 -129105 Yes Yes ISV 92315523  
## 30853 1829258 Yes Yes ISV 92315523  
## 37345 2436412 Yes Yes ISV 92315523  
## 38827 1515528 Yes Yes Services 92315523  
## 78771 1047415 Yes Yes ISV 92315523  
## 84280 1224496 Yes Yes Services 92315523  
## 117685 4919849 Yes Yes Services 92315523  
## 122728 5088071 Yes No Services 453261204

# No missing values but na values in offerReadiness column  
  
which(is.na(partnerSol))  
  
  
partnerSol$OfferStatus = as.factor(partnerSol$OfferStatus)  
partnerSol$OfferReadiness = as.factor(partnerSol$OfferReadiness)  
partnerSol$SolutionKey = as.factor(partnerSol$SolutionKey)  
partnerSol$OfferType = as.factor(partnerSol$OfferType)  
partnerSol$IsAvailableInAppSource = as.factor(partnerSol$IsAvailableInAppSource)  
partnerSol$IsAvailableInMarketplace = as.factor(partnerSol$IsAvailableInMarketplace)  
partnerSol$IsTransactableInMarketplace = as.factor(partnerSol$IsTransactableInMarketplace)  
partnerSol$IsISVConnectSolution = as.factor(partnerSol$IsISVConnectSolution)  
partnerSol$IsIncentiveEligible = as.factor(partnerSol$IsIncentiveEligible)  
  
#Data set does not contain any outliers as each column is categorical or   
#Identification number categorical variables are consistent across table and   
#don't need string manipulation  
  
  
#Duplicate rows not found in data.  
#Code identifies there is no duplicate values. True if there are no duplicates   
#otherwise False.  
dim(partnerSol)[1]==dim(unique(partnerSol))[1]  
  
#Solution key serves as a primary key. Number of rows of unique solution keys   
#match total number of rows in table  
length(partnerSol$SolutionKey) == dim(partnerSol)[1]

#Contains data mapping SolutionKeys to SolutionAreaKeys. Is important for   
#seeing Solution keys that have not had success yet along with area.  
  
  
#Using is.na function which returns True in case of missing values and False   
#otherwise and which() function to return positions of missing values.   
#No missing values were found.  
  
which(is.na(partnerSolArea))  
  
  
#Data set does not contain any outliers as each column is Identification number   
#convert keys to factors  
  
partnerSolArea$SolutionKey = as.factor(partnerSolArea$SolutionKey)  
partnerSolArea$SolutionAreaKey = as.factor(partnerSolArea$SolutionAreaKey)  
  
  
  
#Duplicate rows not found in data.  
#Code identifies there is no duplicate values. True if there are no duplicates   
#otherwise False.  
dim(partnerSolArea)[1]==dim(unique(partnerSolArea))[1]  
  
  
#A composite key for this table of Solution Key and Solution Area Key is   
#appropriate as solution key doesn't uniquely identify table alone.

#not found with function but found with summary after converting to date  
which(is.na(partnerSpec))  
  
#Convert categorical variables to factors and StartFiscalMonthID to date  
partnerSpec$PartnerKey = as.factor(partnerSpec$PartnerKey)  
partnerSpec$Workload = as.factor(partnerSpec$Workload)  
partnerSpec$AdvProgramKey = as.factor(partnerSpec$AdvProgramKey)  
partnerSpec$EnrollmentStatusKey = as.factor(partnerSpec$EnrollmentStatusKey)  
partnerSpec$StartFiscalMonthID = as.Date(partnerSpec$StartFiscalMonthID,format= "%Y-%m-%d")  
  
  
#Final Structure  
str(partnerSpec)  
  
  
#Check for duplicates, no duplicates with composite key of all columns  
#Get rid of duplicates of composite key   
#PartnerKey,AdvProgramKey, and EnrollmentStatusKey  
  
  
CleanedPartnerSpec = partnerSpec[!duplicated(partnerSpec[,c(1,3,5)]),]  
  
dim(CleanedPartnerSpec)[1]==dim(unique(CleanedPartnerSpec))[1]  
dim(CleanedPartnerSpec)

Note: Changed Nulls to 0 for Active Usage in BIZAPP

# changing data types for appropriate columns  
azure$FiscalMonthID <- as.factor(azure$FiscalMonthID)  
azure$PartnerKey <- as.factor(azure$PartnerKey)  
azure$CustomerKey <- as.factor(azure$CustomerKey)  
azure$SolutionAreaKey <- as.factor(azure$SolutionAreaKey)  
  
# check structure of data  
str(azure)

## 'data.frame': 138203 obs. of 5 variables:  
## $ FiscalMonthID : Factor w/ 6 levels "379","380","381",..: 6 2 5 2 4 3 2 2 4 3 ...  
## $ PartnerKey : Factor w/ 130 levels "-9999","1014822",..: 10 48 48 51 28 68 8 62 62 53 ...  
## $ CustomerKey : Factor w/ 6730 levels "606756007","630377459",..: 1746 3023 5693 1745 2492 2714 5536 4759 6320 1682 ...  
## $ SolutionAreaKey: Factor w/ 24 levels "1","2","5","8",..: 20 14 8 8 19 2 20 8 8 20 ...  
## $ Revenue : num 343.7 121.5 25.9 2305.1 51.5 ...

summary(azure)

## FiscalMonthID PartnerKey CustomerKey SolutionAreaKey  
## 379:20650 1018532:22007 676959435: 870 44 :38960   
## 380:21567 1202533:17035 676952790: 787 19 :20446   
## 381:22537 1020955:16850 676953028: 661 42 :16470   
## 382:23632 1121211: 9895 676953446: 630 27 :10876   
## 383:24522 1142665: 9213 676952733: 607 45 :10555   
## 384:25295 1023126: 7410 676953132: 580 26 : 7312   
## (Other):55793 (Other) :134068 (Other):33584   
## Revenue   
## Min. : -3049   
## 1st Qu.: 120   
## Median : 551   
## Mean : 11479   
## 3rd Qu.: 3003   
## Max. :5460048   
##

# check for missing values  
sum(is.na(azure))

## [1] 0

#change column name to prevent errors  
colnames(bizapp)[which(names(bizapp) == "Active.Usage")] <- "ActiveUsage"  
# changing data types for appropriate columns  
bizapp$FiscalMonthID <- as.factor(bizapp$FiscalMonthID)  
bizapp$PartnerKey <- as.factor(bizapp$PartnerKey)  
bizapp$CustomerKey <- as.factor(bizapp$CustomerKey)  
bizapp$SolutionAreaKey <- as.factor(bizapp$SolutionAreaKey)  
bizapp$ActiveUsage <- as.integer(bizapp$ActiveUsage)  
  
# check structure of data  
str(bizapp)

## 'data.frame': 199050 obs. of 5 variables:  
## $ FiscalMonthID : Factor w/ 5 levels "380","381","382",..: 2 2 2 3 3 3 5 2 5 5 ...  
## $ PartnerKey : Factor w/ 5080 levels "-108550","-102207",..: 142 60 1402 220 4 3760 266 77 4 1192 ...  
## $ CustomerKey : Factor w/ 32089 levels "608436186","608463157",..: 18124 24000 15039 31143 21459 14793 23905 29650 11779 328 ...  
## $ SolutionAreaKey: Factor w/ 14 levels "59","61","62",..: 8 8 8 8 8 8 8 8 8 8 ...  
## $ ActiveUsage : int 12 17 4 23 10 12 2 14 3 12 ...

summary(bizapp)

## FiscalMonthID PartnerKey CustomerKey SolutionAreaKey   
## 380:38059 -9999 :48776 676953028: 383 69 :186333   
## 381:39121 1166415:13383 676996872: 286 62 : 4470   
## 382:40173 1121211:13341 676952864: 276 70 : 2564   
## 383:40553 1023126:10074 676952894: 226 66 : 1631   
## 384:41144 1018532: 9118 676956058: 175 68 : 1406   
## 1540400: 7947 677017184: 161 64 : 968   
## (Other):96411 (Other) :197543 (Other): 1678   
## ActiveUsage   
## Min. : 0.00   
## 1st Qu.: 1.00   
## Median : 3.00   
## Mean : 27.98   
## 3rd Qu.: 12.00   
## Max. :43505.00   
##

# View(bizapp)  
# pairs(bizapp)  
# head(bizapp)  
  
# check for missing values  
sum(is.na(bizapp))

## [1] 0

# changing data types for appropriate columns  
modwrk$FiscalMonthID <- as.factor(modwrk$FiscalMonthID)  
modwrk$PartnerKey <- as.factor(modwrk$PartnerKey)  
modwrk$CustomerKey <- as.factor(modwrk$CustomerKey)  
modwrk$SolutionAreaKey <- as.factor(modwrk$SolutionAreaKey)  
colnames(modwrk)[which(names(modwrk) == "Active.Usage")] <- "ActiveUsage"  
  
# check structure of data  
str(modwrk)

## 'data.frame': 806946 obs. of 5 variables:  
## $ FiscalMonthID : Factor w/ 5 levels "380","381","382",..: 2 2 1 3 3 5 2 5 3 2 ...  
## $ PartnerKey : Factor w/ 71 levels "-9999","1014822",..: 1 42 1 1 1 1 1 29 1 1 ...  
## $ CustomerKey : Factor w/ 131355 levels "608274577","608327200",..: 8900 124029 102578 102016 57563 41324 69314 116287 50360 2133 ...  
## $ SolutionAreaKey: Factor w/ 10 levels "73","74","75",..: 5 10 5 5 5 5 5 1 5 5 ...  
## $ ActiveUsage : int 11 0 1 0 0 0 5 185 0 0 ...

summary(modwrk)

## FiscalMonthID PartnerKey CustomerKey SolutionAreaKey   
## 380:148131 -9999 :583001 676996872: 597 79 :621332   
## 381:155283 1121211: 62540 676952864: 510 80 : 54111   
## 382:161259 1018532: 44416 676952473: 334 74 : 52253   
## 383:167925 1359050: 27826 676953028: 292 77 : 37367   
## 384:174348 1023126: 22871 676952894: 272 73 : 20331   
## 1018856: 13948 683174138: 270 150 : 9870   
## (Other): 52344 (Other) :804671 (Other): 11682   
## ActiveUsage   
## Min. : -256   
## 1st Qu.: 0   
## Median : 2   
## Mean : 1057   
## 3rd Qu.: 39   
## Max. :2153654   
##

# check for missing values  
sum(is.na(modwrk))

## [1] 0

str(consumption)

## 'data.frame': 17442 obs. of 4 variables:  
## $ CustomerKey : int 677621078 676954032 677620575 683841861 676953047 683173853 683864722 683841742 676952831 676969667 ...  
## $ Workload : chr "BIZAPP" "BIZAPP" "MODWRK" "AZURE" ...  
## $ PartnerOneKey: int 1022389 1021608 1118995 4655346 5400562 1416163 1016027 4640959 4504602 1018747 ...  
## $ Consumption : num 6.87 5.24e+01 3.27e+03 5.13e+03 1.08e+07 ...

#Convert categorical variables to factors  
consumption$CustomerKey = as.factor(consumption$CustomerKey)  
consumption$Workload = as.factor(consumption$Workload)  
consumption$PartnerOneKey = as.factor(consumption$PartnerOneKey)  
  
#No duplicates found.  
consumption[duplicated(consumption),]

## [1] CustomerKey Workload PartnerOneKey Consumption   
## <0 rows> (or 0-length row.names)

#summary statistics  
str(customer)  
summary(customer)  
  
#checking for null values, there are none in any column  
sum(is.na(customer))  
  
#all key variables, so changed to factors  
customer$CustomerKey = as.factor(customer$CustomerKey)  
customer$SalesTerritoryID = as.factor(customer$SalesTerritoryID)  
customer$VerticalKey = as.factor(customer$VerticalKey)  
  
#checking for duplicate rows, no duplicates  
dim(customer)[1] == dim(unique(customer))[1]

#activity key is an identification number, so the appropriate data type should be changed to factor instead of numeric  
activity$ActivityKey = as.factor(activity$ActivityKey)  
  
#summary statistics  
str(activity)  
summary(activity)  
  
#checking for null values, there are none in either column  
sum(is.na(activity))  
  
#checking for duplicate rows  
dim(activity)[1] == dim(unique(activity))[1]

#each column is an identification number except FiscalMonthID, therefore this is the only column capable of producing outliers. The columns that are identification numbers are converted to factors.  
individual\_activity$PartnerKey = as.factor(individual\_activity$PartnerKey)  
individual\_activity$IndividualKey = as.factor(individual\_activity$IndividualKey)  
individual\_activity$ActivityKey = as.factor(individual\_activity$ActivityKey)  
  
#summary statistics  
str(individual\_activity)  
summary(individual\_activity)  
  
#checking for null values, there are none in any column  
sum(is.na(individual\_activity))  
  
#checking for duplicate rows  
dim(individual\_activity)[1] == dim(unique(individual\_activity))[1]  
  
#getting rid of the duplicate rows. individual key and activity key are the composite key for this table so they are condition of uniqueness  
C\_individual\_activity = individual\_activity %>% distinct(ActivityKey, IndividualKey, .keep\_all = TRUE)  
  
#checking if the cleaned table matches the unique values of the original  
dim(C\_individual\_activity)[1] == dim(unique(individual\_activity))[1]

CustomerWins$CustomerKey = as.factor(CustomerWins$CustomerKey)  
CustomerWins$PartnerKey = as.factor(CustomerWins$PartnerKey)  
CustomerWins$SolutionKey = as.factor(CustomerWins$SolutionKey)  
CustomerWins$SolutionAreaKey = as.factor(CustomerWins$SolutionAreaKey)  
CustomerWins$DealTypeKey = as.factor(CustomerWins$DealTypeKey)  
CustomerWins$FiscalMonthID = as.factor(CustomerWins$FiscalMonthID)  
  
CustomerWins = CustomerWins[!duplicated(CustomerWins),]

## 

## Data Reshaping

Created a SQLlite database of files so could manipulate and join tables easier later on in making plots. Hid some of code for joins for graphs.

con <- dbConnect(RSQLite::SQLite(), dbname = "Microsoft")  
dbWriteTable(con, "partner", CleanedPartner,overwrite=TRUE)  
dbWriteTable(con, "partnerSolution", partnerSol,overwrite=TRUE)  
dbWriteTable(con, "partnerSolutionArea", partnerSolArea,overwrite=TRUE)  
dbWriteTable(con, "customerWins", CustomerWins,overwrite=TRUE)  
dbWriteTable(con, "perfMODWRK", modwrk,overwrite=TRUE)  
dbWriteTable(con, "perfBIZAPP", bizapp,overwrite=TRUE)  
dbWriteTable(con, "perfAZURE", azure,overwrite=TRUE)  
dbWriteTable(con, "solutionAreaMap", SolutionAreaMap,overwrite=TRUE)  
dbWriteTable(con, "activity", activity,overwrite=TRUE)  
dbWriteTable(con, "consumption", consumption,overwrite=TRUE)  
dbWriteTable(con, "partnerSpec", CleanedPartnerSpec,overwrite=TRUE)  
dbWriteTable(con, "customer", customer,overwrite=TRUE)  
dbWriteTable(con, "indivAct",individual\_activity ,overwrite=TRUE)  
dbWriteTable(con, "CusKeyParKeyCount", CusKeyParKeyCount,overwrite=TRUE)

# 

# Key Metrics/Stats

## AZURE

### Sum of Revenue - $1,586,472,292

### Average Revenue

Fiscal Month ID - $264,412,049

Solution Area Key - $66,103,012

Customer Key - $235,731

Partner Key - $12,203,633

## BIZAPP

### Sum of Active Users - 5,568,436

### Average Active Users

Fiscal Month ID – 1,113,687

Solution Area Key – 397,745

Customer Key – 174

Partner Key – 1,096

## MODWRK

### Sum of Active Users - 853,051,765

### Average Active Users

Fiscal Month ID – 170,610,353

Solution Area Key – 85,305,176

Customer Key – 6,494

Partner Key – 12,014,814

# Partner

### Number of partners- 156,438

### Number of ProgramTiers - 5

summary(CleanedPartner[,2:4])

## IsManaged IsCosellReadyPartner PartnerSegment   
## No :154771 No :153553 ISV : 1934   
## Yes: 1677 Yes: 2895 N/A :153934   
## Services: 580

# Partner Solution

### Number of solutions-6958

summary(partnerSol[,2:8])

## OfferReadiness OfferStatus OfferType IsAvailableInAppSource  
## 2:4895 Live :5524 819786134 :3343 No :3224   
## 3:2063 N/A : 791 152591786 :1994 Yes:3734   
## Not Live: 643 2147483647: 409   
## 1852531623: 329   
## 1430932346: 256   
## 1388461353: 234   
## (Other) : 393   
## IsAvailableInMarketplace IsTransactableInMarketplace IsISVConnectSolution  
## No :3003 No :6444 No :6467   
## Yes:3955 Yes: 514 Yes: 491   
##   
##   
##   
##   
##

## PartnerSpecialization

summary(partnerSpec[,2:5])

## Workload AdvProgramKey StartFiscalMonthID EnrollmentStatusKey  
## AZURE :46865 430375863 :20062 Min. :2018-12-29 0:59307   
## BIZAPP: 5250 928063080 : 5584 1st Qu.:2021-08-30 2: 4597   
## MODWRK:13384 143075323 : 5175 Median :2021-10-28 3: 684   
## 189506201 : 4119 Mean :2021-10-16 4: 16   
## 2068489875: 3887 3rd Qu.:2021-12-11 5: 16   
## 1610187822: 3088 Max. :2022-01-24 6: 879   
## (Other) :23584 NA's :65056

## Activity

## Percentages of counts of each type:

Exam 28.15%

Track 21.20%

Course 20.03%

Assessment 17.56%

Certification 13.06%

The only notable column is Type, which is a categorical variable. So I wanted to see the percentage of the counts of each type. Exam appeared the most at 28.15%, and Certification appearing the least at 13.06%.

## Individual Activity

### Top 3 Partner Key: frequency

10209551:35.18%

11212111: 9.84%

10209581:3.34%

This was another table with nearly all key values, with the only exception being FiscalMonthID. There is not much to say about that column specifically because the it represents a period of time and values lie relatively close together. There was grouping done on each key value, and the counts of each are shown in descending order. It is also notable that the most frequently appearing Partner Key appeared about 35.18% of the time.

# Customer

## Top 3 Sales Territory ID: frequency of customers

25806:72.4%

3010779:4.8%

19204:1.8%

## Top 3 Vertical key: frequency of customers

167511991:50.5%

1666091001: 4.6%

858914936: 3.5%

This was another table with all key values for data. Because of this, grouping was an effective way to explore the frequencies of each key. For Sales Territory ID and Vertical Key both, there was one in each that dominated their respective frequencies. The most common Sales Territory ID made up about 72.4% of all Sales Territory IDs. The most common Vertical Key made up about 50.5% of all Vertical Keys.

# Partner and Customer relationship statistics

In these tables we get top 10 counts of partners and customers that have worked together before as well as counts of top 10 solutions and workload for specfic customer. There also is a table of looking in to which partner workload combination have occurred the most.

## `summarise()` has grouped output by 'CustomerKey'. You can override using the  
## `.groups` argument.  
## `summarise()` has grouped output by 'CustomerKey', 'SolutionKey'. You can  
## override using the `.groups` argument.  
## `summarise()` has grouped output by 'PartnerKey'. You can override using the  
## `.groups` argument.  
## `summarise()` has grouped output by 'Workload'. You can override using the  
## `.groups` argument.

## # A tibble: 10 × 4  
## # Groups: CustomerKey, SolutionKey [10]  
## CustomerKey SolutionKey Workload Count  
## <fct> <fct> <fct> <int>  
## 1 683841872 23479 AZURE 44  
## 2 676953072 10889 AZURE 37  
## 3 676953003 17546 AZURE 36  
## 4 676952688 10889 AZURE 35  
## 5 676952864 17546 AZURE 30  
## 6 683173905 5933 MODWRK 30  
## 7 676952790 10889 AZURE 27  
## 8 676952924 23479 AZURE 26  
## 9 676953319 23479 AZURE 26  
## 10 676965287 17546 AZURE 26

## # A tibble: 10 × 3  
## # Groups: CustomerKey [9]  
## CustomerKey PartnerKey Count  
## <fct> <fct> <int>  
## 1 683841872 1020353 47  
## 2 676953072 4458787 37  
## 3 676953003 1104812 36  
## 4 676952688 4458787 35  
## 5 676952864 1104812 30  
## 6 677011693 4210355 30  
## 7 683173905 1399212 30  
## 8 676952688 1432842 27  
## 9 676952790 4458787 27  
## 10 676952924 1020353 26

## # A tibble: 10 × 3  
## # Groups: PartnerKey [10]  
## PartnerKey Workload Count  
## <fct> <fct> <int>  
## 1 1104812 AZURE 5168  
## 2 4210355 MODWRK 2209  
## 3 4504602 AZURE 2039  
## 4 1023309 AZURE 1985  
## 5 4458787 AZURE 1870  
## 6 1020661 AZURE 1323  
## 7 4922182 AZURE 1215  
## 8 1169329 Unknown 1143  
## 9 2107902 BIZAPP 1113  
## 10 1854867 AZURE 1094

# Top 10 partners, individualKeys, and activity

## PartnerKey CountPartKey  
## 1 10209551 552712  
## 2 11212111 154700  
## 3 10209581 52490  
## 4 13590501 48132  
## 5 15608861 41250  
## 6 14161631 37584  
## 7 10185321 36060  
## 8 11426651 33606  
## 9 10148221 32982  
## 10 11642051 27590

## IndividualKey CountIndKey  
## 1 73148956 7670  
## 2 2352646 7434  
## 3 147866266 6962  
## 4 78874586 6490  
## 5 91270126 5700  
## 6 133220496 5546  
## 7 135828876 5428  
## 8 148431196 5428  
## 9 94384086 5074  
## 10 83115356 4956

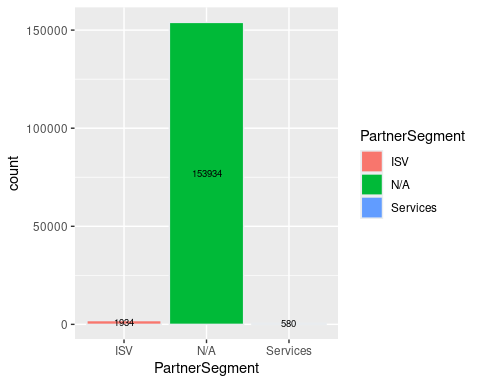
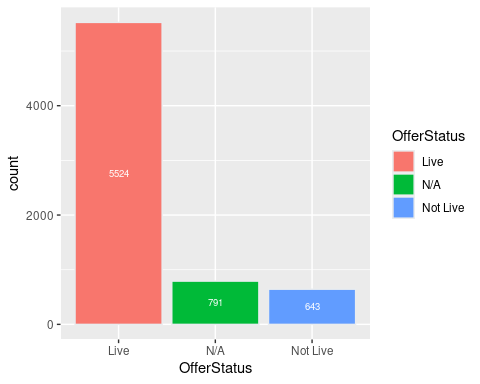
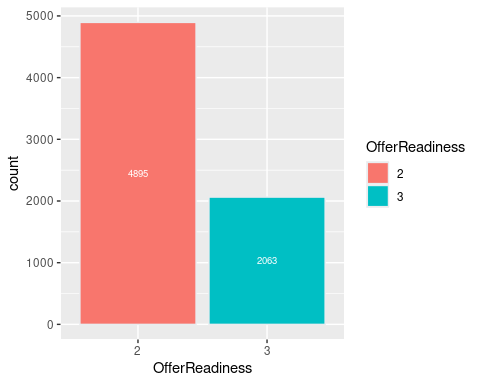
## ActivityKey CountActKey  
## 1 5096 33847  
## 2 26736 30568  
## 3 24636 27480  
## 4 24646 27479  
## 5 25942 24972  
## 6 25952 24971  
## 7 35996 22238  
## 8 25876 21227  
## 9 26846 19139  
## 10 37136 19139

# 

# Graphs

## Partner distribution of attributes

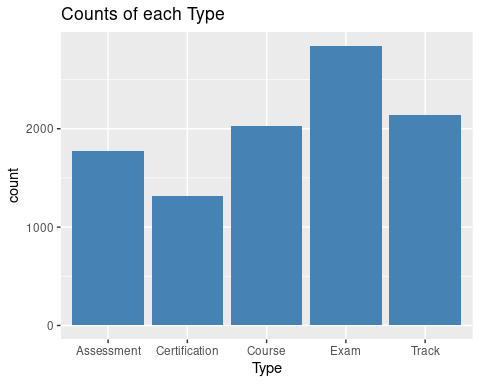
These graphs you can see amount of managed partner is alot less than not managed. More partner solutions are not as ready in offer readiness. Also, more offers are live compared to not live or NA. Plus, A lot of partners are not labeled as ISV partner or service partner.



# 

# Distribution of partner training activity types

## # A tibble: 5 × 3  
## Type count percent   
## <chr> <int> <formttbl>  
## 1 Exam 2845 28.15%   
## 2 Track 2143 21.20%   
## 3 Course 2025 20.03%   
## 4 Assessment 1775 17.56%   
## 5 Certification 1320 13.06%

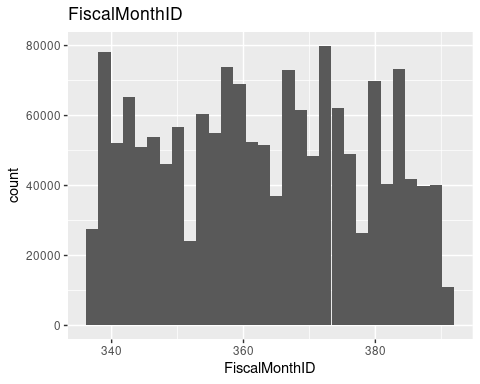


# 

# Histogram: Counts of total partner activity training based on month.

## [1] TRUE

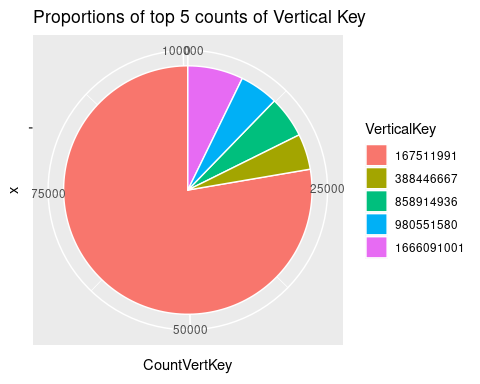
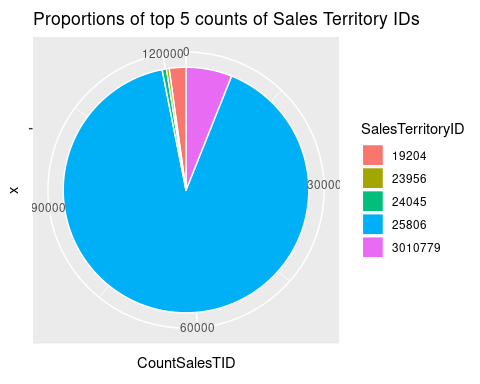
## `stat\_bin()` using `bins = 30`. Pick better value with `binwidth`.



# 

# Pie charts: Proportion of SalesTerritoryIDs and Vertical IDs

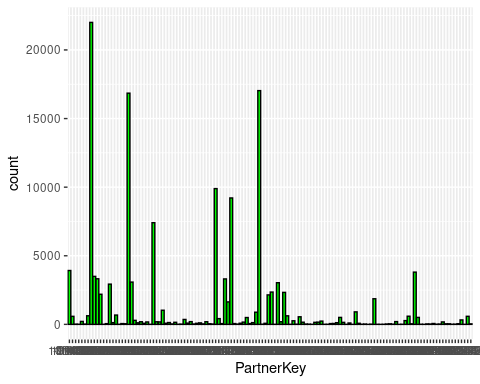
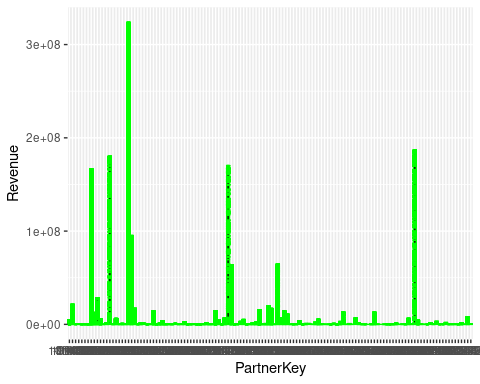
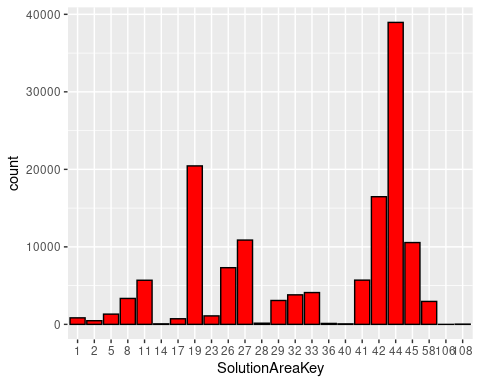
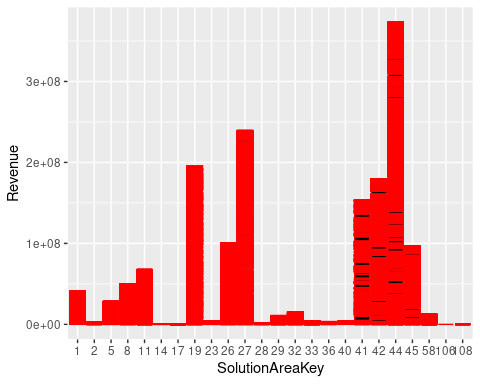
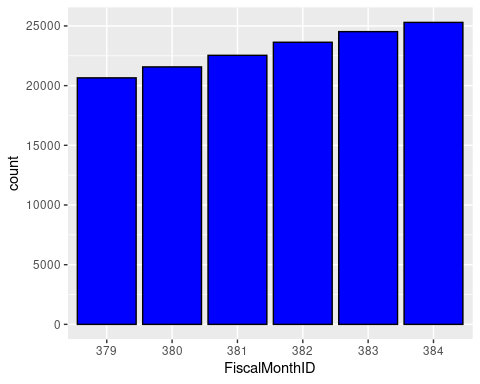
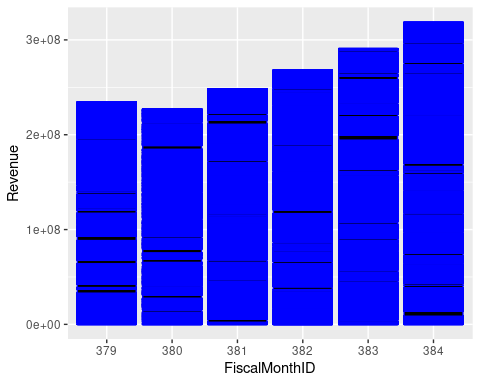
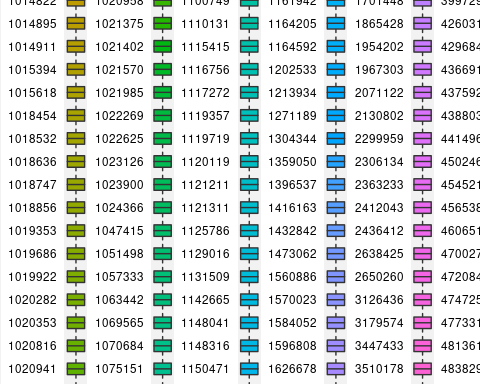
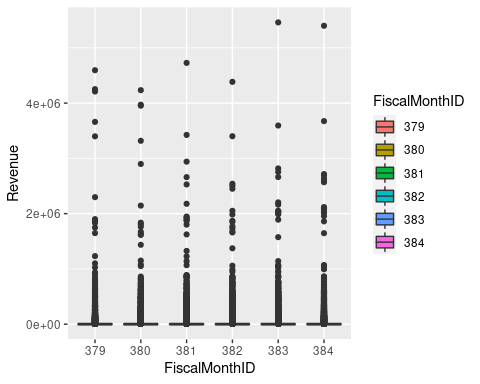
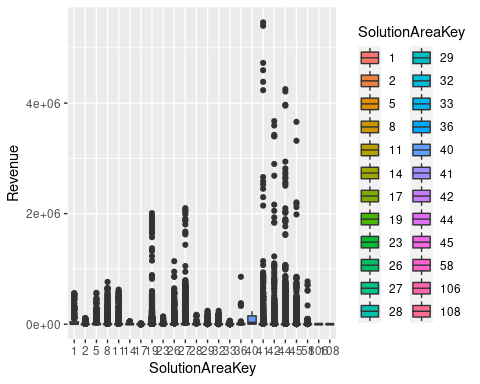
From these graphs we can see a certain sales territory dominates customer base. Also, the same is true looking at the Vertical Key.



# Performance graphs

### AZURE

Many outliers exist in the data, and I would say the only usefulness of the box plots that were generated was to determine the existence of outliers. The histogram that gives us the most insight is the SolutionAreaKey category. This very clearly shows the areas that generate the most revenue. The aggregated tables show in descending order the keys that provide the most revenue.



## Category x  
## 6 384 318471629  
## 5 383 290763194  
## 4 382 267865736  
## 3 381 248057455  
## 1 379 234173662  
## 2 380 227140617

## Category x  
## 20 1020955 323738031  
## 112 4813611 186726120  
## 14 1019922 180098470  
## 52 1131509 170163996  
## 8 1018532 166522451  
## 21 1020958 95184652

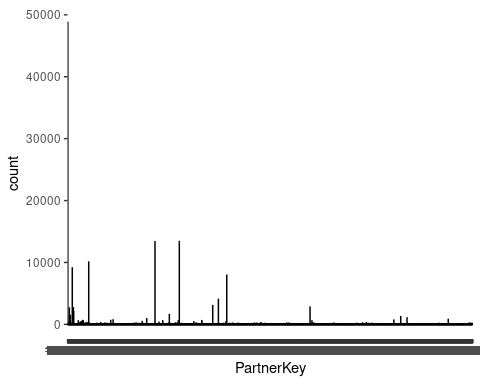
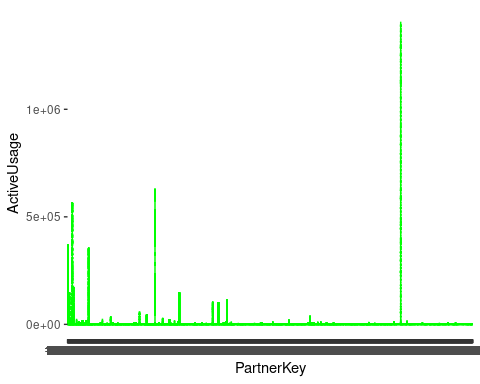
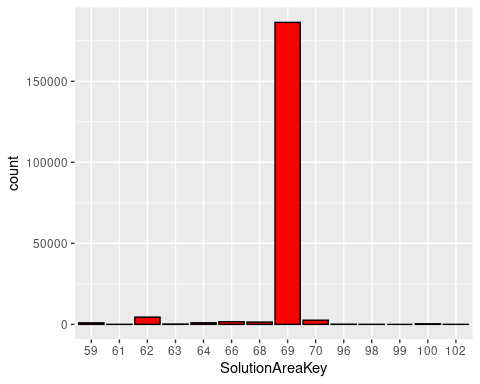
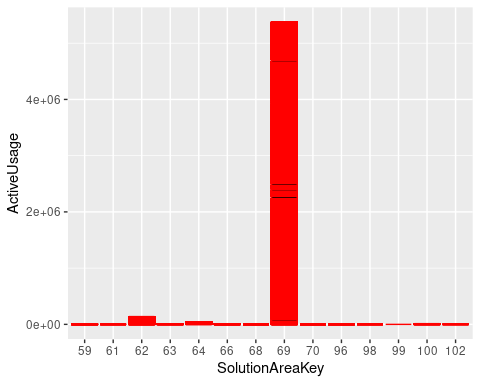
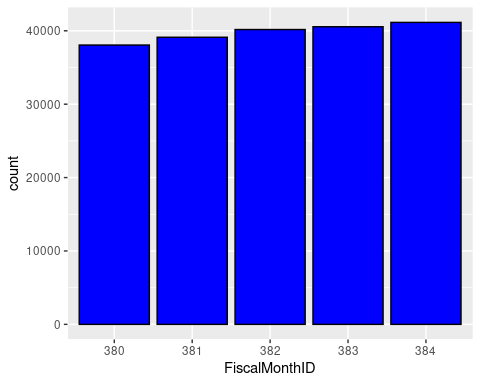
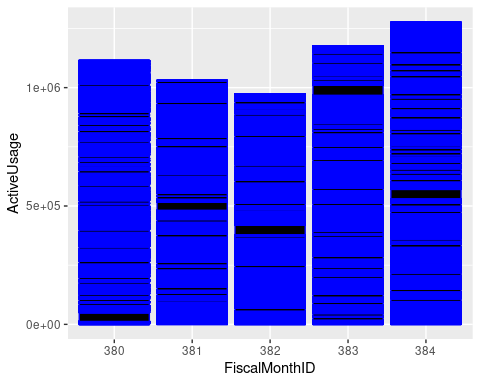
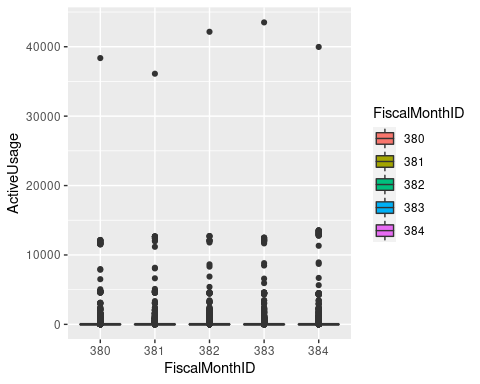
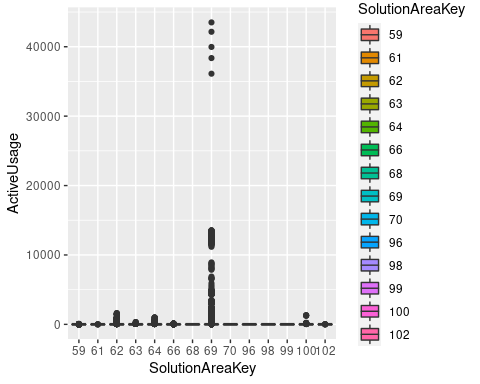
## Category x  
## 3604 676953072 160727846  
## 3834 676956058 80380914  
## 3677 676953446 68807234  
## 3565 676952878 53188637  
## 3547 676952790 51235710  
## 4113 676969015 49216092

## Category x  
## 20 44 372941629  
## 11 27 239392937  
## 8 19 195493988  
## 19 42 179527537  
## 18 41 153625025  
## 10 26 100718696

### 

### BIZAPP

Much like the AZURE dataset, this dataset has a large number of outliers and this is once again confirmed in the boxplots. Also much like the AZURE dataset, the histogram that is most useful is the one using SolutionAreaKey. There is a clear standout among the group. The aggregated tables show in descending order the keys that contain the most active users.



## Category x  
## 5 384 1275464  
## 4 383 1175833  
## 1 380 1114316  
## 2 381 1029894  
## 3 382 972929

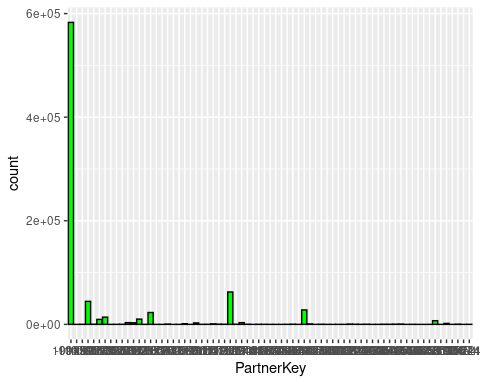
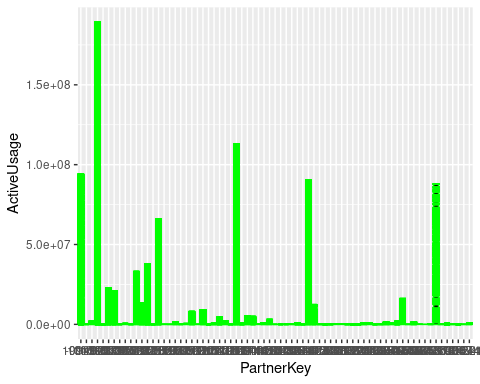
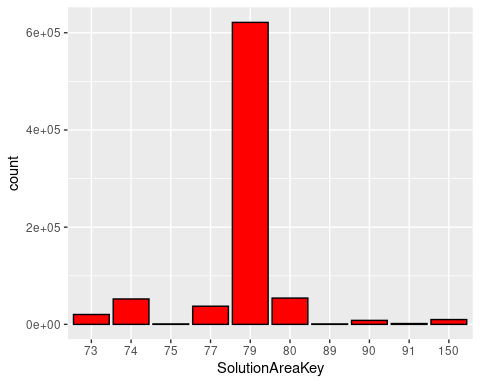
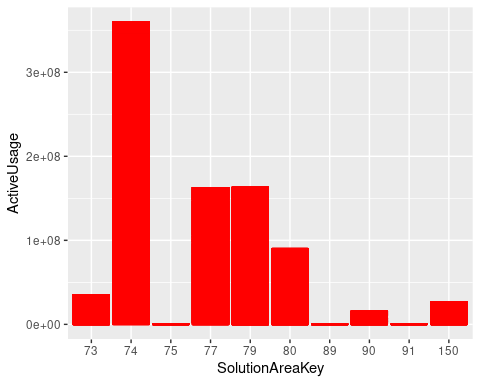
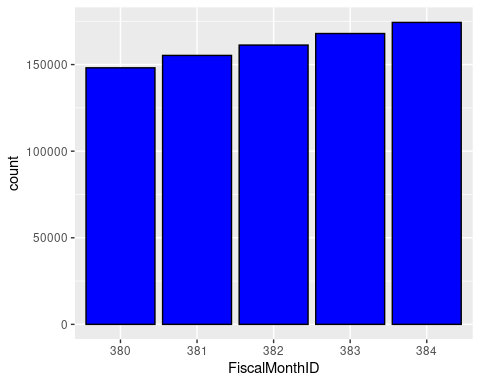
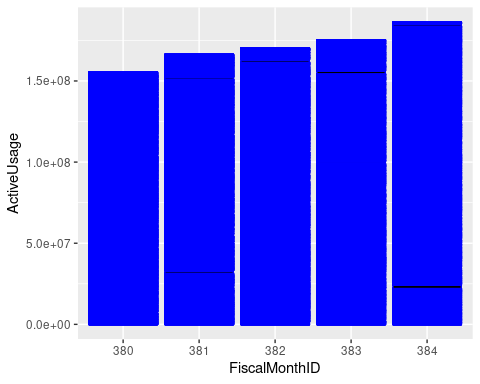
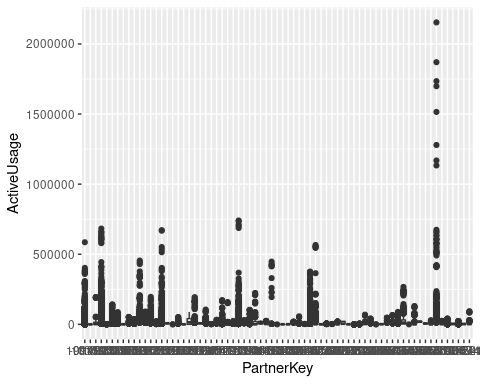
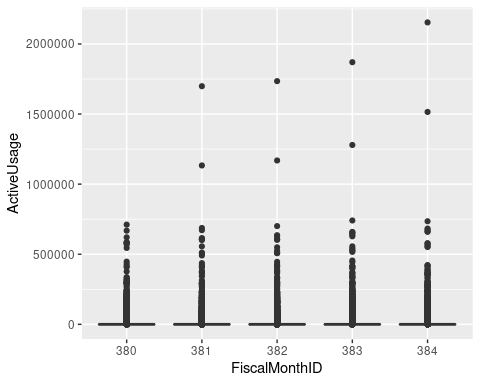
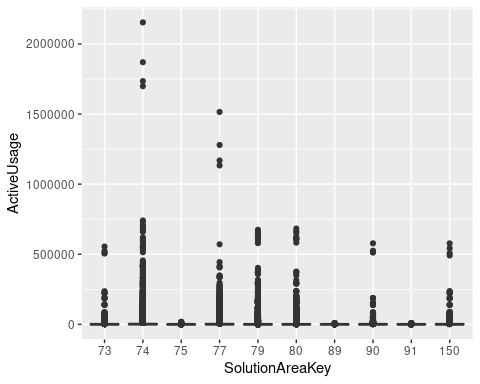
## Category x  
## 4178 4813611 1403985  
## 1097 1121211 627047  
## 60 1018532 563097  
## 4 -9999 367517  
## 266 1023126 353533  
## 74 1018747 176371

## Category x  
## 22083 676956058 966277  
## 22013 676955603 427617  
## 22063 676955876 225734  
## 21417 676952956 83958  
## 28825 680466662 62744  
## 21632 676953464 62086

## Category x  
## 8 69 5369911  
## 3 62 136348  
## 5 64 37741  
## 13 100 11851  
## 6 66 7883  
## 4 63 3851

### MODWRK

Much like the AZURE dataset, this dataset has a large number of outliers and this is once again confirmed in the boxplots. Also much like the AZURE dataset, the histogram that is most useful is the one using SolutionAreaKey. There are some clear standouts among the group. The aggregated tables show in descending order the keys that contain the most active users.



## Category x  
## 5 384 186097747  
## 4 383 175210061  
## 3 382 170145169  
## 2 381 166267885  
## 1 380 155330903

## Category x  
## 4 1018532 189076228  
## 29 1121211 113098888  
## 1 -9999 94028068  
## 42 1359050 90245109  
## 65 4813611 88018229  
## 15 1023126 65983724

## Category x  
## 114647 676956058 32455802  
## 113703 676952858 12064858  
## 114616 676955876 11092453  
## 116278 676973242 9959915  
## 114026 676953464 9416731  
## 125598 680466662 8790598

## Category x  
## 2 74 359355499  
## 5 79 162895598  
## 4 77 162566751  
## 6 80 90074040  
## 1 73 35058895  
## 10 150 26965400

# Joins

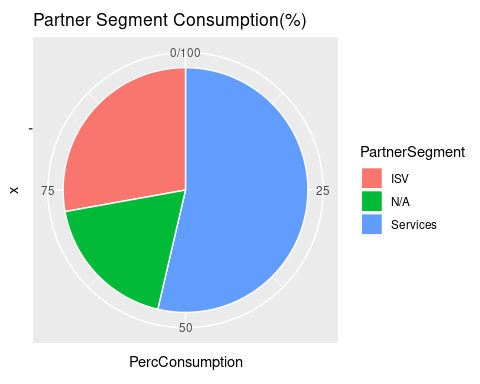
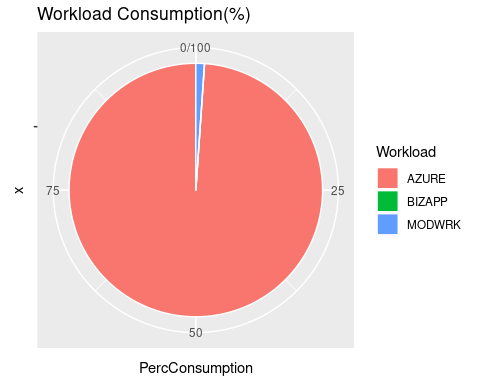
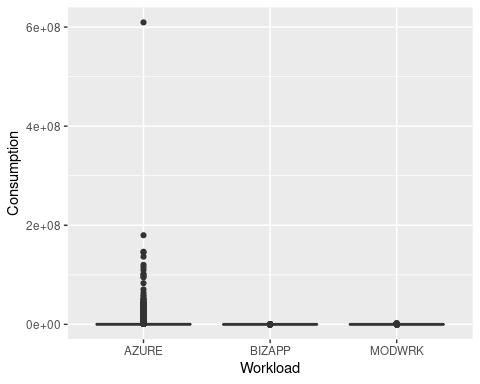
SELECT Workload,ROUND(sum(Consumption)\*100/(SELECT sum(Consumption)   
 FROM consumption as c, partner as p  
 WHERE p.PartnerKey = c.PartnerOneKey),2) as PercConsumption  
FROM consumption as c, partner as p  
WHERE p.PartnerKey = c.PartnerOneKey  
GROUP BY Workload  
ORDER BY PercConsumption DESC

SELECT PartnerSegment,ROUND(sum(Consumption)\*100/(SELECT sum(Consumption)   
 FROM consumption as c, partner as p  
 WHERE p.PartnerKey = c.PartnerOneKey), 2) as PercConsumption   
FROM consumption as c, partner as p  
WHERE p.PartnerKey = c.PartnerOneKey  
GROUP BY PartnerSegment  
ORDER BY PercConsumption DESC

# 

# Consumption graphs

In consumption we segmented based on workload and partner segment. Based on the graphs we can see that AZURE dominates that amount of consumption and as far as partner segmentation, services are more prominent.



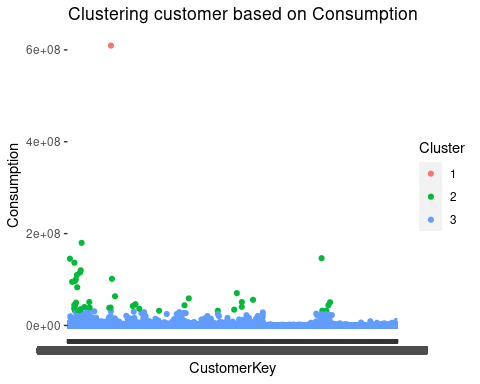
## Category x  
## 1 AZURE 9824886953  
## 3 MODWRK 110527571  
## 2 BIZAPP 565944

## Category x  
## 301 1018532 1370476698  
## 406 1023126 953767016  
## 812 1121211 886106764  
## 1131 1359050 270407830  
## 311 1018747 241459958  
## 1262 1692954 179904638

## Category x  
## 1197 676955876 610146100  
## 389 676953072 179903668  
## 7001 683173866 146573394  
## 64 676952203 145470374  
## 189 676952688 137050276  
## 364 676953033 120542876

# Clustering graph

Clustered customers based on revenue and can see clear customer outlier that is pulling in a lot more revenue then other customers as well as groups customers based on how much consumption they are providing.



# Partners vs. Performance grouping by enrollment status

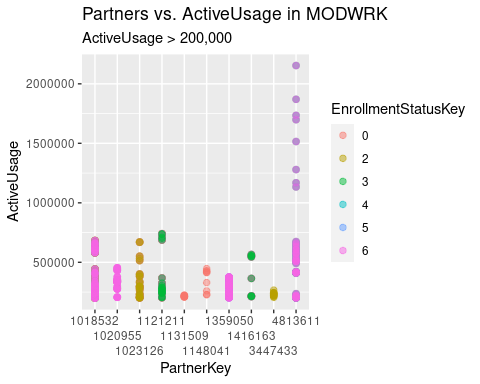
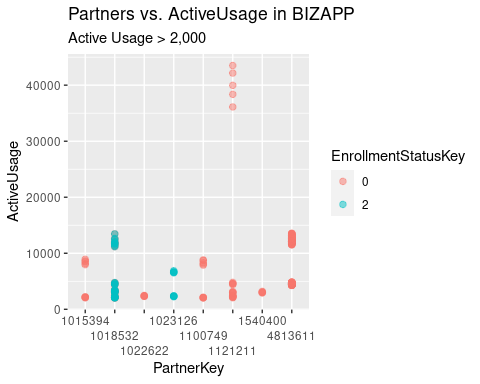
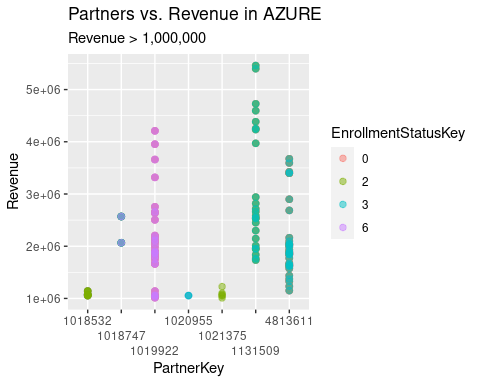
Made these plots in attempt to distinguish partners that haven’t enrolled yet but are making progress (0-6) and see how much revenue they are pulling in.

### sql

SELECT Revenue,EnrollmentStatusKey,ps.PartnerKey,AdvProgramKey  
FROM (SELECT \* FROM perfAZURE WHERE Revenue>=1000000) as pa,   
(SELECT \* FROM partnerSpec WHERE Workload = "AZURE") as ps  
WHERE ps.PartnerKey = pa.PartnerKey

SELECT ActiveUsage,EnrollmentStatusKey,ps.PartnerKey,AdvProgramKey  
FROM (SELECT \* FROM perfBIZAPP WHERE ActiveUsage>2000) as pb,   
(SELECT \* FROM partnerSpec WHERE Workload = "BIZAPP") as ps  
WHERE ps.PartnerKey = pb.PartnerKey

SELECT ActiveUsage,EnrollmentStatusKey,ps.PartnerKey,AdvProgramKey  
FROM (SELECT \* FROM perfMODWRK WHERE ActiveUsage>200000) as pm,   
(SELECT \* FROM partnerSpec WHERE Workload = "MODWRK") as ps  
WHERE ps.PartnerKey = pm.PartnerKey



# Solutions and Customers graph

This shows a tendency for Partners to prefer specific Workloads to win over Customers

