R Code

b=F\_Order\_March25th

library(Hmisc)

cor(b, use="complete.obs", method="spearman")

cor(b, use="complete.obs", method="pearson")

cor(b, use="complete.obs", method="kendall")

SQL Code

--frequency

select [order\_status],

Count( [order\_status] ) AS Frequency

  FROM [dbo].[F\_ORDER]

  GROUP BY [order\_status]

  ORDER BY COUNT([order\_status] ) DESC

--AVG and Total REvenue--

select avg([total\_ex\_tax]- [shipping\_cost\_ex\_tax]) as avgrevenue,  sum([total\_ex\_tax]- [shipping\_cost\_ex\_tax]) as totalrevenue, [month], [year], [month\_number]

from [dbo].[D\_DATE] d, [dbo].[F\_ORDER] f

where f.[order\_date\_key]=d.[date\_key]

group by [month], [year], [month\_number]

order by [year], [month\_number]

-- cast to convert decimal to numeric

select cast(avg([total\_ex\_tax]- [shipping\_cost\_ex\_tax])as numeric) as avgrevenue,  cast(sum([total\_ex\_tax]- [shipping\_cost\_ex\_tax]) as numeric) as totalrevenue, [month], [year], [month\_number]

from [dbo].[D\_DATE] d, [dbo].[F\_ORDER] f

where f.[order\_date\_key]=d.[date\_key]

group by [month], [year], [month\_number]

order by [year], [month\_number]

--Drop Time from Date--

select convert (date, [the\_date], 111)

from [dbo].[D\_DATE]

---http://www.w3schools.com/sql/func\_convert.asp---

--Customers with more than one order--

select [customer\_key], count(distinct [order\_ID]) as number\_of\_orders

from [dbo].[F\_ORDER] f, [dbo].[D\_DATE] d

where f.[order\_date\_key]=d.[date\_key]

and [year]=2015

and [order\_status]='Shipped'

group by customer\_key

having count (distinct  [order\_ID])>1

---avg number of orders of all the repeat customers--

select avg(a.number\_of\_orders)

from (select [customer\_key], count(distinct [order\_ID]) as number\_of\_orders

from [dbo].[F\_ORDER] f, [dbo].[D\_DATE] d

where f.[order\_date\_key]=d.[date\_key]

and [year]=2015

and [order\_status]='Shipped'

group by customer\_key

having count (distinct  [order\_ID])>1) a

use kaggle;

go

create table trans\_kag(

id bigint,

chain bigint,

dept bigint,

category bigint,

company bigint,

brand int,

date1 date,

productsize float,

productmeasure varchar(2),

purchasequantity bigint,

purchaseamount float

)

go

select sum([total\_ex\_tax]-[shipping\_cost\_ex\_tax]) as totalrevenue , [month], [year], [month\_number]

from [dbo].[D\_DATE] d, [dbo].[F\_ORDER] f

where f.[order\_date\_key]=d.[date\_key]

--and [store\_credit\_amount] < 0

group by [month], [year], [month\_number]

order by [year], [month\_number]

round and convert to big int

cast(ROUND([purchaseamount],0) as bigint ) as [purchaseamount]

sql round

SELECT ROUND([purchaseamount],0)

from [dbo].[kaggle80]

select

 right('0000' + cast(datepart(year,[date1]) as varchar(4)), 4)

+ right('00' + cast(datepart(month, [date1]) as varchar(2)), 2)

from  [dbo].[kaggle80]

splitting data

smp\_size <- floor(0.70 \* nrow(dat))

train <- dat [1:smp\_size, ]

test <- dat [ (smp\_size+1): nrow(dat), ]

smp\_size

nrow(test)

nrow(train)

nrow(dat)

library(forecast)

# Map 1-based optional input ports to variables

dataset1 <- train # class: data.frame

dataset2 <- test # class: data.frame

seasonality<-1

labels <- as.numeric(dataset1$data)

timeseries <- ts(labels,frequency=seasonality)

model <- ets(timeseries)

numPeriodsToForecast <- ceiling(max(dataset2$time)) - ceiling(max(dataset1$time))

numPeriodsToForecast <- max(numPeriodsToForecast, 0)

forecastedData <- forecast(model, h=numPeriodsToForecast)

forecastedData <- as.numeric(forecastedData$mean)

#output <- data.frame(cbind(dataset2,forecastedData))

output <- data.frame(time=dataset2$time,forecast=forecastedData)

data.set <- output

# Select data.frame to be sent to the output Dataset port

maml.mapOutputPort("data.set");