**בס"ד**

**Home Task**

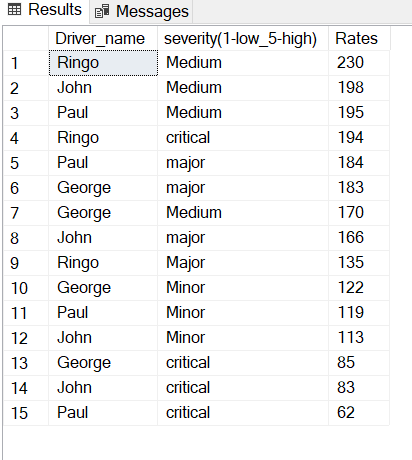
**Name: Gil Biton**

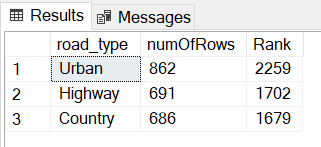
**Id : 316512987**

**All the used scripts**

**Test Questions :**

1. **The driver that needs to change the way he characterize issues is Ringo , Because all his severity ranks is medium and higher . he characterizes in a strict way**

****

1. **The most challenging road type is Urban ,It's get the highest rank (2.62)** ****
2. **The country with the worst performance is Italy . It's get the highest rank (3.16)**

**תמונה שמכילה שולחן

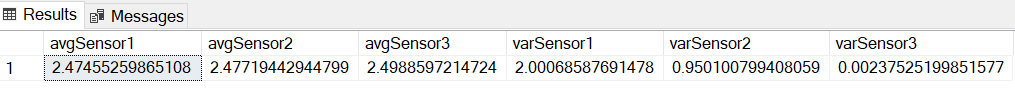
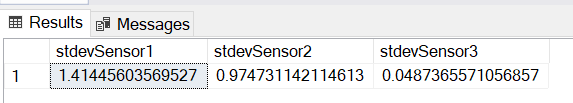
התיאור נוצר באופן אוטומטי**

1. **The illumination with the best results is Dusk , It's get the lowest rank(2.22)**

תמונה שמכילה שולחן

התיאור נוצר באופן אוטומטי

**Sensor Questions :**

**** ****

**1)Sensor 3 ,its AVG is the closest to 2.5 ,** **This gives a best estimate of a value 2.5**

**2)Sensor 1 , because it's var is the highest** **And the results are distributed uniformly and continuously, therefore it is more difficult to conclude from this about the value 2.5 .In Addition The variance indicates to us the dispersion of the samples in relation to the parameter being estimated, the higher the variance, the greater the inaccuracy in estimating the parameter**

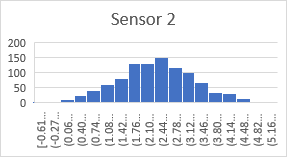
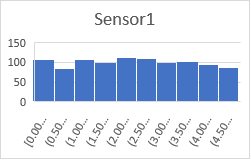
**3)Sensor 1 – 1.41445 , Sensor 2 – 0.97473 , Sensor 3 -0.04873**

**4) There is 2 options for sensor distribution - continuous uniform and normal .uniform distribution means every value in the distribution range is equally likely to occur whether normal distribution looks like "bell" and means that values near the center are more likely to occur .** **With the help of a graph we can tell where the distribution may have come from**

**Sensor 1 - probably came from a continuous uniform distribution, because the graph is uniformly distributed**

**Sensor 2 -** **probably came from a normal distribution, because there is a "bell" shape, the area around the parameter is higher than the other parameters**

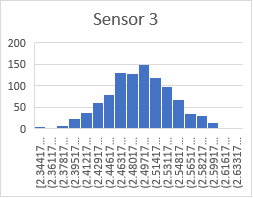
**Sensor 3 - probably came from a normal distribution, because there is a "bell" shape, the area around the parameter is higher than the other parameters**



1. **Sensor 1 -2.849929**

**Sensor 2 - 2.6618**

**Sensor 3 -2.49933**



**Parser Questions :**

1. **At row 41 the speed is much higher than all the data (370.9 after normalization) And it might indicate a mistake . This happened because the data has a sequence of "7D-5E" entered in the speed calculation, instead it should have been 7E .**
2. **The wrong fcs is at row 36 , it's should be 3D and not 4D**

**תמונה שמכילה טקסט

התיאור נוצר באופן אוטומטי**

**Code Scripts :**

**RMS calculation - in excel**

**=SQRT((SUMSQ(A2:A1001)/COUNTA(A2:A1001)))**

**=SQRT((SUMSQ(B2:B1001)/COUNTA(B2:B1001)))**

**=SQRT((SUMSQ(C2:C1001)/COUNTA(C2:C1001)))**

**Sql Server (I imported the excel file into sql):**

use HomeTask

--Test Results Questions

--Question 1

SELECT Driver\_name ,[severity(1-low\_5-high)],[Rates]=Count(\*)

FROM Test\_Results

GROUP BY Driver\_name ,[severity(1-low\_5-high)]

ORDER BY [Rates] DESC

--Question 2

Select road\_type, numOfRows=COUNT(\*), [Rank]=(1\*(select count(\*) FROM Test\_Results as T WHERE [severity(1-low\_5-high)] IN ('Minor') AND T.road\_type=TS.road\_type group by road\_type,[severity(1-low\_5-high)])

+2\*(select count(\*) FROM Test\_Results as T WHERE [severity(1-low\_5-high)] IN ('Medium') AND T.road\_type=TS.road\_type group by road\_type,[severity(1-low\_5-high)])

+3\*(select count(\*) FROM Test\_Results as T WHERE [severity(1-low\_5-high)] IN ('Major') AND T.road\_type=TS.road\_type group by road\_type,[severity(1-low\_5-high)])

+4\*(select count(\*) FROM Test\_Results as T WHERE [severity(1-low\_5-high)] IN ('Critical') AND T.road\_type=TS.road\_type group by road\_type,[severity(1-low\_5-high)]))

FROM Test\_Results as TS

GROUP BY road\_type

ORDER BY [Rank] DESC

--Question 3

SELECT COUNTRY ,numOfRows=Count(\*),[Rank]=(1\*ISNULL((select count(\*) FROM Test\_Results as T WHERE [severity(1-low\_5-high)] IN ('Minor') AND T.COUNTRY=TS.COUNTRY group by COUNTRY,[severity(1-low\_5-high)]),0)

+2\*(select count(\*) FROM Test\_Results as T WHERE [severity(1-low\_5-high)] IN ('Medium') AND T.COUNTRY=TS.COUNTRY group by COUNTRY,[severity(1-low\_5-high)])

+3\*(select count(\*) FROM Test\_Results as T WHERE [severity(1-low\_5-high)] IN ('Major') AND T.COUNTRY=TS.COUNTRY group by COUNTRY,[severity(1-low\_5-high)])

+4\*(select count(\*) FROM Test\_Results as T WHERE [severity(1-low\_5-high)] IN ('Critical') AND T.COUNTRY=TS.COUNTRY group by COUNTRY,[severity(1-low\_5-high)]))

FROM Test\_Results as TS

GROUP BY COUNTRY

ORDER BY [RANK] DESC

--Question 4

SELECT Illumination ,numOfRows=COUNT(\*),[Rank]=(1\*(select count(\*) FROM Test\_Results as T WHERE [severity(1-low\_5-high)] IN ('Minor') AND T.Illumination=TS.Illumination group by Illumination,[severity(1-low\_5-high)])

+2\*(select count(\*) FROM Test\_Results as T WHERE [severity(1-low\_5-high)] IN ('Medium') AND T.Illumination=TS.Illumination group by Illumination,[severity(1-low\_5-high)])

+3\*(select count(\*) FROM Test\_Results as T WHERE [severity(1-low\_5-high)] IN ('Major') AND T.Illumination=TS.Illumination group by Illumination,[severity(1-low\_5-high)])

+4\*(select count(\*) FROM Test\_Results as T WHERE [severity(1-low\_5-high)] IN ('Critical') AND T.Illumination=TS.Illumination group by Illumination,[severity(1-low\_5-high)]))

FROM Test\_Results as TS

GROUP BY Illumination

ORDER BY [RANK]

--Sensor Questions

SELECT avgSensor1 =AVG(Series1) ,avgSensor2=AVG(Series2) ,avgSensor3= AVG(Series3) ,varSensor1=VAR(Series1) ,varSensor2=VAR(Series2),varSensor3=VAR(Series3)

FROM Sensors

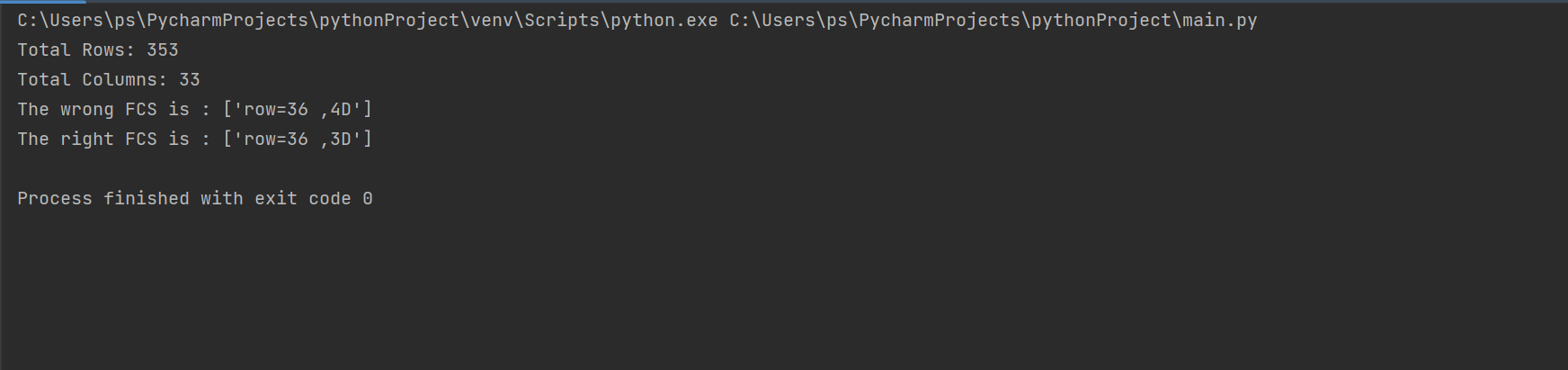
SELECT stdevSensor1=STDEV(Series1),stdevSensor2=STDEV(Series2),stdevSensor3=STDEV(Series3)

FROM Sensors

**Python (I imported the excel file into python):**

import openpyxl  
  
def findFCS(PythonStub):  
  
 sheet\_obj = sheet['Parser Q']  
 row = sheet\_obj.max\_row #row length  
 column = sheet\_obj.max\_column #col length  
 print("Total Rows:", row)  
 print("Total Columns:", column)  
  
 right\_fcs=[]  
 wrong\_fcs = []  
 for j in range(2, row): #a loop that goes through all the rows  
 isChanges = False #indicates changes rows  
 sum = 0  
 for i in range(2, 21): #a loop through all the col ,the range is the header and data bytes range  
 cell\_obj = sheet\_obj.cell(row=j, column=i)  
 print(cell\_obj.value, end=" ") #cell value  
 if(check7E(sheet,j,i)==True) : # function that checks if there is "7D" and then "5E"  
 isChanges=True  
 sum+=int("7E",16)  
 continue  
 elif(check7E(sheet,j,i-1)==True): #skip 7D-5E col  
 continue  
 elif (check7D(sheet, j, i) == True): #function that checks if there is "7D" and then "5D"  
 isChanges = True  
 sum += int("7D", 16)  
 continue  
 elif (check7D(sheet, j, i - 1) == True):#skip 7D-5D col  
 continue  
 else :  
 sum += int(cell\_obj.value, 16) #convert hex2dec  
 temp = hex(255 - (sum % 256))[2:].upper() #fcs formula  
 num=Convert(sum,temp) #function that converts values like 3 to 03 for comparsion  
 print("FCS to hex is" + " " +num)  
 if(isChanges==False): #condition for case that 7D-5E or 7D-5D at the end of data stream  
 if(check7E(sheet,j,21)==True):  
 if(num!="7E"): ##if 7D-5E instead of FCS val then compare to 7E  
 wrong\_fcs.append('row=' + str(j) + " ," + sheet\_obj.cell(row=j, column=21).value)  
 right\_fcs.append('row=' + str(j) + " ," + temp)  
 elif(check7D(sheet,j,21)==True) :  
 if(num!="7D"):  
 wrong\_fcs.append('row=' + str(j) + " ," + sheet\_obj.cell(row=j, column=21).value)  
 right\_fcs.append('row=' + str(j) + " ," + temp)  
 elif (num != sheet\_obj.cell(row=j, column=21).value):  
 wrong\_fcs.append('row=' + str(j) + " ," + sheet\_obj.cell(row=j, column=21).value )  
 right\_fcs.append('row=' + str(j) + " ," + temp)  
 else:  
 if(num != sheet\_obj.cell(row=j, column=22).value):  
 wrong\_fcs.append('row=' + str(j) + " ," + sheet\_obj.cell(row=j, column=21).value)  
 right\_fcs.append('row=' + str(j) + " ," + temp)  
 print('The FCS val is' + " " + str(255 - (sum % 256)))  
 print("The wrong FCS is :", wrong\_fcs)  
 print("The right FCS is :", right\_fcs)  
  
def Convert (sum,temp) :  
 if (255 - (sum % 256)< 16):  
 return '0' + temp  
 else :  
 return temp  
def check7E(PythonStub,j,i) :  
 sheet\_obj=sheet['Parser Q']  
 if(sheet\_obj.cell(row=j, column=i).value=="7D" and sheet\_obj.cell(row=j, column=i+1).value=="5E"):  
 return True  
 else:  
 return False  
def check7D(PythonStub,j,i) :  
 sheet\_obj=sheet['Parser Q']  
 if(sheet\_obj.cell(row=j, column=i).value=="7D" and sheet\_obj.cell(row=j, column=i+1).value=="5D"):  
 return True  
 else:  
 return False  
  
if \_\_name\_\_ == '\_\_main\_\_':  
 path = "Copy of Data\_analysis\_Q - V2.4.xlsx"  
 sheet=openpyxl.load\_workbook(path)  
  
 findFCS(sheet)

**Output :**

****