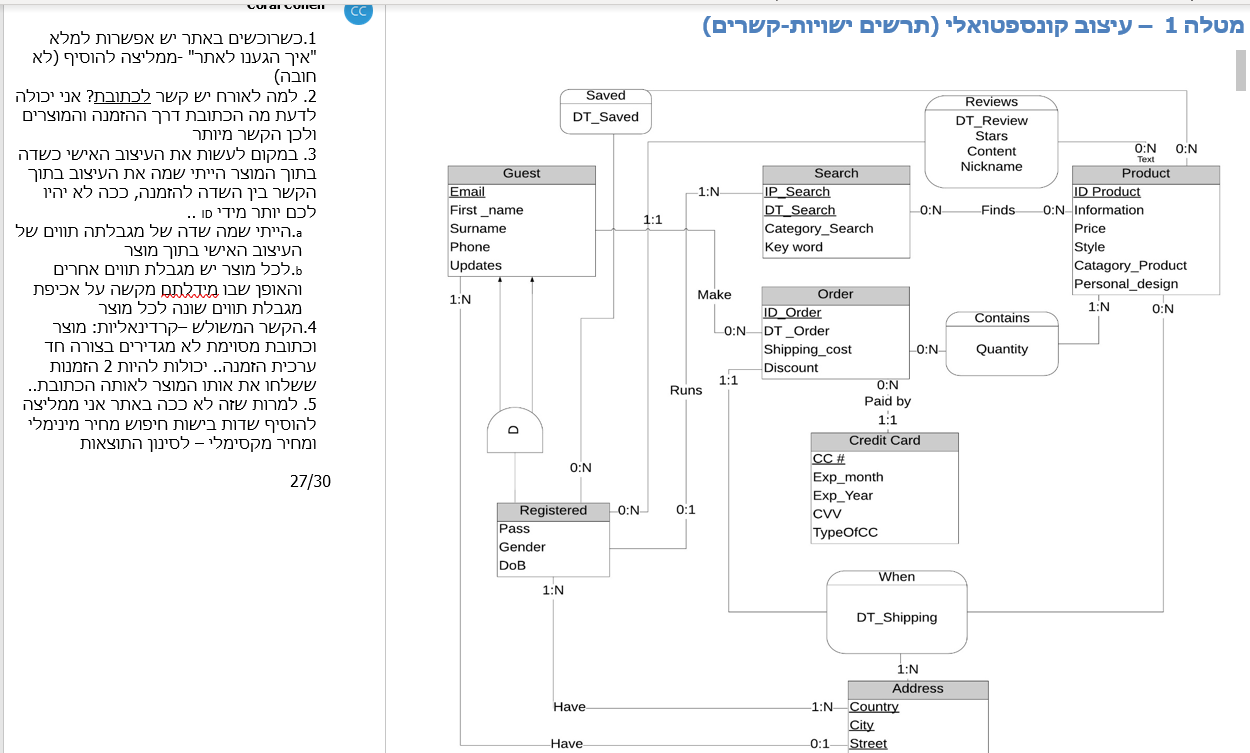
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| מס' קבוצה | אתר | | | תאריך הגשה |
| 01 | <https://www.personalcreations.com/> | | | 20/06/2020 |
| חברי הצוות - מספרי ת.ז | | | | |
| 205731748 | | 315625301 | 305470957 | |

## פרויקט בסיסי נתונים – חלק ג'

## מטלת חובה מקדימה – תיקון ה-ERD והעשרת בסיס נתונים

## (קנס של עד 20% לציון על ביצוע לא ראוי)

ERD עם הערות מחלק ב':



ERD מתוקן:

תמונה שמכילה טקסט

התיאור נוצר באופן אוטומטי

## מודל טבלאי עם הערות מחלק ב':

## מודל טבלאי מתוקן:

## מטלה 1 (25%) – שאילתות

# שתי שאילתות SELECT ללא קינון 5%

השאילתה הבאה שולפת את עשרת המוצרים שהוזמנו עם כמות יחידות שגדולה/שווה מממוצע כמות היחידות שהוזמנו בהזמנה אחת כאשר מתייחסים לשנת 2019 ומעלה.

התרחיש העסקי שעבורו השאילתא מספקת מענה הוא כאשר מנהלי הביניים ירצו לבדוק איזה מוצר הוזמן בכמות גדולה ביחס לשאר המוצרים שהוזמנו בשנת 2019 ומעלה, על מנת שיוכלו לבחון האם המוצר מוזמן באופן סיטונאי ועל כן יש לבחן את תקן המלאי שלהם למוצרים הללו, כיוון שאולי יש צורך להגדיל/להקטין אותו.

select top 10 con.ID\_Product, con.ID\_Order, con.Quantity

from Orders as o join Containments as con on o.ID\_Order = Con.ID\_Order join

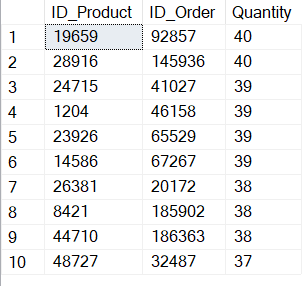
Products AS P ON con.ID\_Product = p.ID\_Product

where year(o.DT\_Order) > 2018

group by con.ID\_Product, con.Quantity, con.ID\_Order

having con.Quantity>=avg(con.Quantity)

order by 3 desc



השאילתה הבאה שולפת את המוצרים שקיבלו ביקורות נמוכות (3 כוכבים ומטה) מפי גברים בשנת 2019 ואילך.

למנהלים כדאי לדעת אילו מוצרים קיבלו ביקורות שלילית מפי גברים בשנים האחרונות מכיוון שכך עובדי החברה יוכלו לבצע ניתוח נקודתי באוכלוסיית הגברים, להבין למה הביקורות שליליות ולאחר מכן לגבש אסטרטגיה לשיפור המוצרים בעיניי אוכלוסיית הגברים.

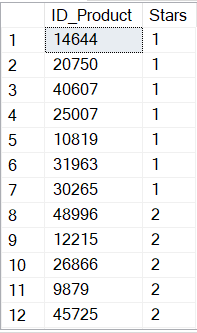
select p.ID\_Product, rev.Stars

from Registered AS R JOIN Reviews AS REV ON R.Email = REV.Email JOIN Products AS P ON

REV.ID\_Product = P.ID\_Product

where YEAR(REV.DT\_Review)> 2018 AND R.Gender = 'M' and rev.Stars<=3

ORDER BY 2 asc

(לא העתקנו את כל הרשומות מהפלט כי יש 19 כאלו)

# שתי שאילתות SELECT מקוננות 10%

השאילתא הבאה פולטת את המוצרים מקטגוריית SALE, אשר חלים עליהם הנחה בגובה 10% ומעלה, מחירם קטן מהממוצע ושנמכרו בשנה האחרונה.

ההצדקה העסקית לבחירת השאילתא היא כדי לבחון לאילו מוצרים יש פרופורציה לא מידתית בין הנחה ביחס למחיר, ז"א אם יש מוצרים שמחירם נמוך מהממוצע ובכל זאת הם מקבלים הנחה גדולה בשיעור 10% ומעלה. זה מועיל לחברה כי כך מקבלי ההחלטות יוכלו להגדיל את הכנסות החברה ע"י צמצום שיעור ההנחה או העלאת מחיר המוצר.

select p.ID\_Product, p.Price, o.Discount, avgDiscount= (select avg (Discount) from Orders)

from Orders as o join Shipping\_Times as s on o.ID\_Order=s.ID\_Order

join Products as p on p.ID\_Product=s.ID\_Product

where p.Category\_Product='SALE' and o.Discount>=10 and year(getdate())-year(DT\_Shipping)<=1

group by p.ID\_Product, p.Price, o.Discount

having p.Price<(select avg (price) from Products)

order by 2 desc



השאילתא הבאה חושפת מי הם האנשים ששמרו מוצרים בעגלת הקניות אך לא רכשו אותם מסיבה כלשהי אך עדיין רכשו מוצר/ים אחר/ים.

ההצדקה העסקית היא לתת כלי לאנשי המכירות שיעזור להם לעודד רכישות ע"י יצירת קשר עם האנשים המדוברים. המטרה היא לשכנע אותם לחזור ולרכוש את המוצרים שהם חיפשו, באופן שיווקי-סמוי כמו שליחת מיילים עם קטלוג שמכיל את המוצרים שחופשו. מבחינתנו, על אף שלקוח עשוי לא לרצות לקבל עדכונים שוטפים במייל, עלינו בכל זאת לשלוח תזכורות על מוצרים שנשמרו מכיוון שהם עלולים היו להישכח בעגלה וגם מכיוון שזהו כלי ייעודי לאנשי המכירות לעזור להכנסות החברה.

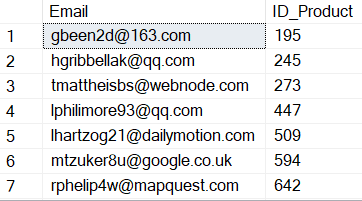
select r.Email,s.ID\_Product

from Registered as r join saved as s on r.Email=s.Email

where r.Email not in (select s.Email from Shipping\_Times as sh join orders as o on sh.ID\_Order=o.ID\_Order

where sh.ID\_Order=s.ID\_Product and o.Email=s.Email)

order by 2



(יש נתונים נוספים בפלט האמיתי)

**שתי שאילתות מקוננות תוך שימוש במרכיבים נוספים 10%**

השאילתא הבאה מעדכנת את מחירי המוצרים תחת קטגורית 'FOR KIDS'.

המטרה העסקית היא לעדכן את המחירים של המוצרים שמחירם מעל הממוצע כדי לדוגמה לתת הנחה של 5% לרגל חג.

update PRODUCTS

set Price = PRODUCTS.Price \* 0.95

where PRODUCTS.Price > (select AVG(Price) from PRODUCTS where Category\_Product = 'FOR KIDS')

and Category\_Product = 'FOR KIDS'

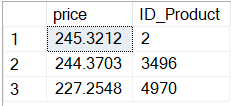
פקודה לבחינת שינויים:

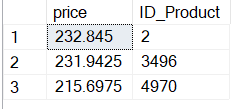
select price, ID\_Product

from Products

where Category\_Product = 'FOR KIDS' and (ID\_Product=2 or ID\_Product= 3496 or ID\_Product=4970)

order by 1 desc

לפני השינוי:

אחרי השינוי:

השאילתא הבאה ממחישה למנהלים את הצורך בנוכחות שווקית חזקה בחגים ומועדים להנחות כמו בלאק פריידי. נראה שבחודש נובמבר נמכרו אפילו מוצרים שהביקורות עליהם שליליות (כוכבים נמוכים מהממוצע) ומחירם גבוה (מעל הממוצע).

המטרה העסקית היא להבין את עוצמת הימים כדוגמת בלאק פריידי ולהעלות לסדר היום של החברה את הצורך בפעילות שיווקית רחבה כדי למכור בכמויות גדולות.

select r.ID\_Product

from Shipping\_Times as st join Reviews as r on st.ID\_Product=r.ID\_Product

group by stars, r.ID\_Product

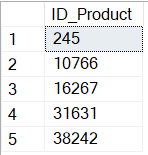
having Stars<(select avg (cast(Stars as real)) from Reviews)

intersect

select p.ID\_Product

from Products as p join findings as f on p.ID\_Product=f.ID\_Product

where month(f.DT\_Search)=11 and Price>(select avg (cast(price as real)) from Products)



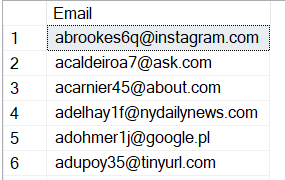
## מטלה 2 (25%) – העשרת בסיס הנתונים

# View 5%

אנשי אבטחת המידע מבצעים אחת לחודש בדיקה אילו סיסמאות הן סיסמאות חלשות, כאשר

סיסמא חלשה היא סיסמא שמכילה פחות מ-8 תווים ו\או מכילה רק אותיות ו\או מכילה רק מספרים.

כמו כן, אנו מעוניינים שאיש האבטחה יקבל רק את המיילים של המשתמשים, ולא את שאר הפרטים עליהם (במיוחד לא את הסיסמא).



CREATE VIEW v\_passValidation AS

SELECT r.Email

FROM Registered as r

WHERE LEN(r.Pass) <=8

OR (r.pass NOT LIKE '%[0-9]%'

AND r.pass NOT LIKE '%[A-Za-z]%')

(יש נתונים נוספים בפלט האמיתי)

בעת הבדיקה יש להריץ את השאילתא הבאה:

SELECT \* FROM v\_passValidation

# פונקציות (Functions) 10%

הפונקציה הראשונה תקבל מס' אשראי ותאריך ותחזיר את עלות העסקה שבוצעה ומתאימה לפרמטרים. ההצדקה העסקית שבמקרה ויש בעיה בעסקה מסוימת שבגללה במקרה שלקוח ירצה החזר כספי, אנשי מתן השירות יוכלו לשלוף את פרטי העסקה בקלות, למצוא את המחיר שהלקוח שילם ולבצע החזר כספי. הסיבה שפונק' מתאימה פה ולא SP היא שיש להכניס פרמטר (מזהה העסקה) כדי ששליפת הנתונים תתאפשר.

CREATE FUNCTION f\_findTransaction (@transID int)

returns real as

begin

declare @price real

SELECT @price=o.Shipping\_Cost+cast((c.Quantity)\*(p.Price)\*(1-o.Discount/100) as real)

FROM products as p join Containments AS c on c.ID\_Product=p.ID\_Product

join orders as o on c.ID\_Order=o.ID\_Order

join Credit\_Cards as ccards on o.CC#=ccards.CC#

WHERE o.ID\_Order=@transID

return @price

end

בעת הבדיקה יש להריץ את השאילתא הבאה:

select price=dbo.f\_findTransaction(30)



הפונקציה השנייה מחזירה מידע על חמשת המוצרים שקיבלו את הדירוג הגבוה בחודשיים האחרונים. המטרה העסקית הנובעת מהפעלת פונקציה זו היא שאנשי השיווק והמכירות בחברה יוכלו לקבל את המידע העדכני ביותר לגבי המוצרים הכי אהובים באתר עבור פרקי זמן קבועים וכתוצאה מכך הם יוכלו לבצע מהלכי שיווק ומכירות שונים אשר יתרמו לחברה עצמה; לדוגמא לפרסם את המוצרים החמים הללו בדף השער באתר החברה או אף לקחת מתוך טבלה זו את המוצר בעל המחיר הנמוך ביותר ולעלות את מחירו בכמה אחוזים לצורך רווח עתידי נוסף.

ההצדקה לעבוד פה עם פונק' ולא עם SP היא מכיוון שניתן לקרוא לפונק' הנ"ל מתוך שאילתא, בניגוד ל-SP. יכול להיות שעובדי החברה ירצו לבצע מניפולציה על טבלת הפלט כמו מה שפירטנו לעיל (למצוא את המוצר העל המחיר הנמוך ביותר) או לדוגמה כאשר ירצו להוסיף סנן של קטגוריה.

CREATE FUNCTION f\_highestRatedProducts ()

RETURNS TABLE

AS RETURN

select top 5 ID\_Product,bestRank=avg(Cast(Stars as real))

from Reviews

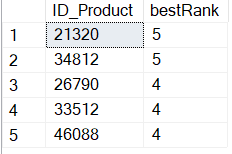
where year(getdate())=year(DT\_Review) and MONTH(getdate())-MONTH(DT\_Review)<=2

group by ID\_Product,Stars

order by Stars desc

בעת הבדיקה יש להריץ את השאילתא הבאה:

select \*from dbo.f\_highestRatedProducts()



# Trigger פשוט 5%

הטריגר שבנינו מתעורר כאשר מתקבלת הזמנה חדשה באתר מפי לקוח רשום. במצב זה נרצה לעדכן את שדה מס' ההזמנות (שיצרנו עבור הטריגר) של הלקוח שביצע את ההזמנה. הרעיון העסקי של ספירת ההזמנות לכל לקוח הוא שבצורה זו נוכל לאתר לקוחות שנוטים לרכוש באופן תכוף מאחרים, כדי שנוכל להציע להם באופן ממוקד הנחות או הטבות בתקווה שהדבר יעודד אותם להמשיך ולבצע רכישות באופן תכוף.

פקודת הוספת העמודה החדשה:

alter table registered add numOfOrders int

נעדכן את הערכים להיות 1 מכיוון שכל הלקוחות הרשומים הזמינו:

update Registered

set numOfOrders=1

והטריגר עצמו:

create trigger t\_updateNumOfOrders

on orders

after insert

as

update Registered

set Registered.numOfOrders=Registered.numOfOrders+

(select count(\*) from inserted where Registered.Email=inserted.Email)

דוגמה להרצת הטריגר:

insert into orders values (12312556,'abannister5o@ucoz.ru',2983651035050350,'2020-06-20 10:34:09',10,10)

# פרוצדורה שמורה ((Stored Procedure פשוטה 5%

בנינו פרוצדורה שנותנת הנחה נוספת עבור הזמנה שמבוצעת ע"י לקוח שנרשם לראשונה לאתר. הפעולה חזרתית ויש להפעיל אותה לעיתים די קרובות, ולכן יש צורך בקוד מובנה שניתן להריץ שוב ושוב- פונק' לא מתאימה פה כי היא צריכה להיקרא מתוך קוד כלשהו. כמו כן, הפעולה שומרת על בטיחות בכך שמאפשרת לאנשי המכירות לבצע שינויים שאינם קשורים לשדות רגישים כמו סיסמה או פרטים אישיים של הלקוח.

CREATE PROCEDURE sp\_specificDiscount @email varchar(50), @additionalDiscount int

AS

UPDATE ORDERS SET orders.Discount=orders.Discount+@additionalDiscount

WHERE orders.Email=@email

הרצה:

EXECUTE SP\_UpdateOrderDiscount 'uhurriondj@phpbb.com', 40

פלט לפני הפרוצדורה:





פלט אחרי הפרוצדורה:





## מטלה 3 (20%) – כלים להצגת נתונים

לוח המחוונים

הלוח מיועד להיות כלי להנהלת החברה ע"מ שיוכלו לקבל החלטות אסטרטגיות שונות.

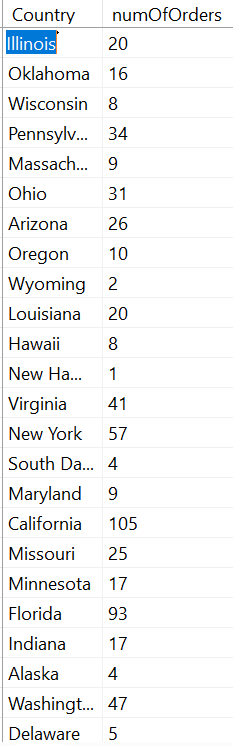
השאלות העסקיות שעליהן יענה הלוח:

1. כמה הזמנות מתבצעות בכל מדינה ובכל עיר?
2. מהי התפלגות ההזמנות לפי קטגוריות?
3. מה ממוצע הכוכבים שמתקבלים בעת ביקורות לפי קטגוריות?
4. כמה רשומים חדשים נמצאים ביחס למדד השנתי?
5. כמה הכנסות היו לנו השנה ביחס לשנים קודמות?
6. כיצד המחיר משפיע על כל הנשאל לעיל?

בעזרת מענה על השאלות הללו, למקבלי ההחלטות יהיה קל יותר לנווט את החברה בתקופה הבאה ע"י כך שיוכלו לבצע פעולות שיווק ממוקדות במדינות שמזמינות פחות, להגביר את פעולות המשלוחים במדינות שמזמינות הרבה, לבחון מחדש נתונים בעזרת סינון לפי מחיר וכו'.

את סעיפים א'-ג' הצגנו באמצעות עזרים ויזואליים שעונים על השאלות באופן מוחשי ומאפשרים השוואה קלה בין נתונים. השתמשו ב-Drill Down כדי לעבור מציר מדינה לציר עיר. יצרנו מד שמראה לנו איפה אנחנו עומדים ביחס ליעד שהוצג בסעיף ד'- כרגע האתר מתאים להכיל עד 5000 רשומים, והמטרה השנתית היא להגיע ל-1000. כדי לענות על סעיף ה' בנינו view ייעודי שנבנה ע"י שאילתא מקוננת, ואף עשינו Drill down כדי לענות על השאלה ברמת החודש. לאחר מכן, הוספנו פילטר מחיר כדי לענות על שאלת סעיף ו'.

הוספנו טבלאות views שיכילו את התשובות לשאלות של לוח המחוונים:



create view v\_OrdersByCountries as

select Country, numOfOrders=count(\*)

from Shipping\_Times

group by country

(יש נתונים נוספים בפלט האמיתי)

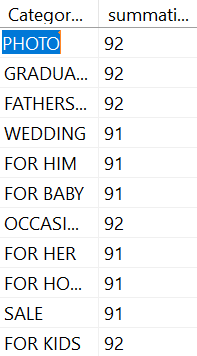
create view v\_CountOfCategoriesByOrders as

select p.Category\_Product, summation=count (\*)

from Orders as o join Shipping\_Times as st on o.ID\_Order=st.ID\_Order

join products as p on st.ID\_Product=p.ID\_Product

group by p.Category\_Product

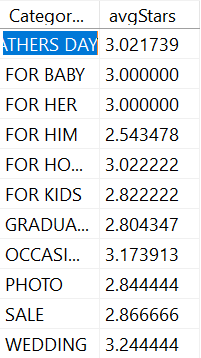


create view v\_starsPerCategory as

select p.Category\_Product, avgStars=avg(Cast(r.Stars as decimal(3,2)))

from Reviews as r join Products as p on r.ID\_Product=p.ID\_Product

group by p.Category\_Product



create view v\_salesPerYear

as

select distinct sos.yearOfOrders ,sumOfSales=sum(sos.sumOfSalesPerOrder)

from (

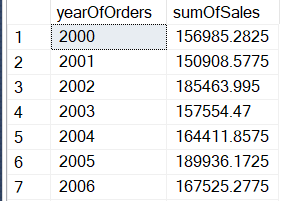
select distinct yearOfOrders=year(o.DT\_Order), sumOfSalesPerOrder=p.Price\*c.Quantity

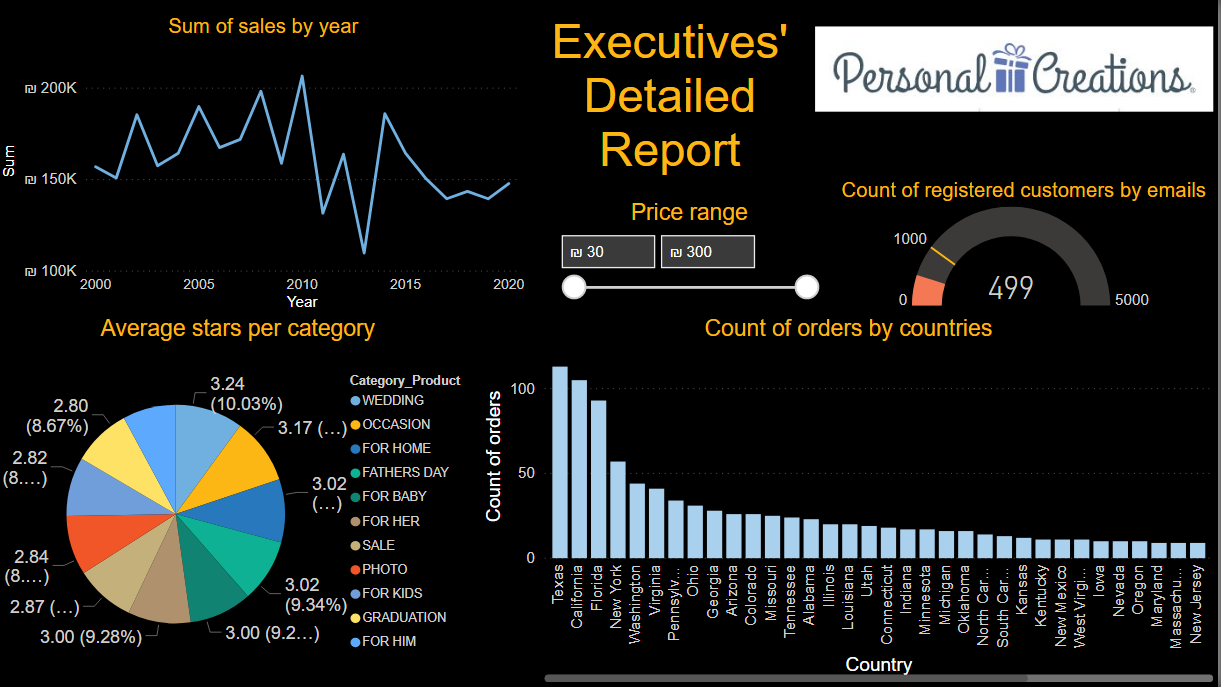
from Products as p join Containments as c on p.ID\_Product=c.ID\_Product

join orders as o on c.ID\_Order=o.ID\_Order

) as sos

group by sos.yearOfOrders

(יש נתונים נוספים בפלט האמיתי)



הדו"ח העסקי

הדו"ח נועד לתת אפשרות ללקוחות פוטנציאליים לקבל מידע כללי על האתר והמוצרים בו לפני הרכישה.

השאלות העסקיות שעליהן יענה הדו"ח:

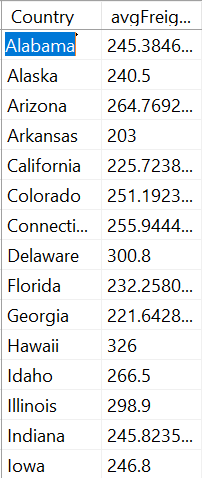
1. מה מחיר המשלוח הממוצע לכל מדינה?
2. מהו המחיר הממוצע עבור כל קטגוריה?
3. איך מתפלגים חיפושי המוצרים בין הקטגוריות?
4. מה המחיר המקס' והמינ' בכל קטגוריה?

המטרה היא שלקוח יוכל לקבל תמונת מצב כללית: אם לקוח מתעניין במוצר או מוצרים ספציפיים הוא יכול לנתח את כדאיות הרכישה שלהם ע"י סננים כלליים של מחירי משלוח ממוצעים, מחירים ממוצעים ופופולריות החיפושים של המוצרים. בנינו אמצעים ויזואלים שיעזרו ללקוח לקבל החלטה על שאלות א'-ג' ובנוסף הוא כמובן יוכל לסנן קטגוריות לפי מחיר. לדוגמה: במידה והלקוח לדוגמה קונה מתנה לאדם אהוב והוא לא בטוח באיזו מתנה כדאי לו לבחור, הוא יוכל לסנן את בחירתו לפי המחיר והחיפושים כמדד לפופולריות. ענינו על שאלת סעיף ד' בעזרת פילטר על הקטגוריות וכרטיסיות מחיר מקס' ומינ'.

הוספנו טבלאות views שיכילו את התשובות לשאלות של הדו"ח העסקי:

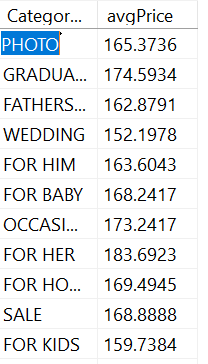
create view v\_avgFreightPerState as

select st.Country, avgFreight=avg(Cast(o.Shipping\_Cost as real))

from Shipping\_Times as st join orders as o on st.ID\_Order=o.ID\_Order

group by st.Country

(יש נתונים נוספים בפלט האמיתי)

create view v\_avgPriceByCategory as

select Category\_Product, avgPrice=avg(Price)

from Products

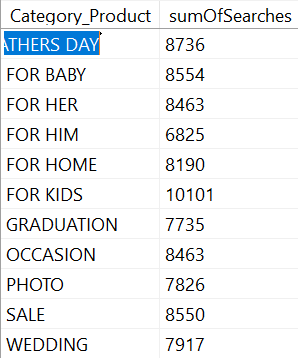
group by Category\_Product

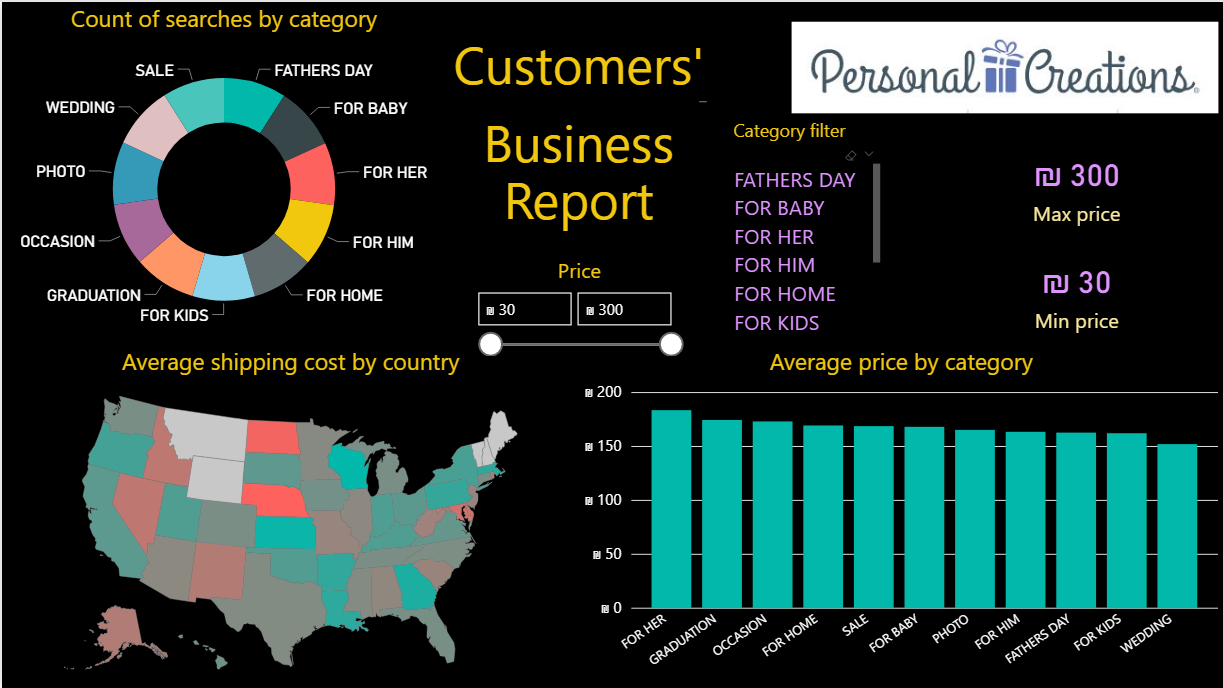
create view v\_searchesPerCategory as

select p.Category\_Product, sumOfSearches=count (\*)

from searches as s join Products as p on s.Category\_Search=p.Category\_Product

group by p.Category\_Product





## מטלה 4 (צבירה - עד 35%) – יישומים מתקדמים

## כלי מורכב (Stored Procedure או Trigger) 10%

המטרה שלנו היא למצוא את העיר בכל מדינה שבה הכי כדאי לעבוד עם חברת שליחויות מקומית כדי לזרז את זמן המשלוח- כמו שאמזון עובדת בארץ עם חברות שליחויות. משיקולי תקציב מדובר בעיר אחת בכל מדינה. כדאי למצוא את העיר הכי כדאית לשת"פ בחרנו את המדד הבא: אנו רוצים את העיר שבה כמות ההזמנות היא הגבוהה ביותר מבין כל הערים באותה מדינה, וגם שסך כמות ההזמנות בה ביחס לסך ההזמנות של המדינה גדול מ-50%.

הפונק' הראשונה מחשבת את סך ההזמנות בכל עיר:

create function f\_totalPerCity ()

returns table

as

return

select stt.Country, stt.City, total=count(\*)

from Shipping\_Times as stt join orders as o on stt.ID\_Order=o.ID\_Order

group by stt.Country, stt.City

הפונק' השנייה מוצאת את הערים הכי פופולריות (העיר שבה ביצעו הכי הרבה הזמנות) בכל מדינה תוך הישענות על הפלט מהפונק' הראשונה:

create function f\_getTopCities ()

returns table

as

return

select distinct tt.Country, tt.City, tt.total

from f\_totalPerCity() as tt

where tt.total>=all(select dbo.f\_totalPerCity.total from f\_totalPerCity() where dbo.f\_totalPerCity.country=tt.country)

group by tt.Country, tt.City, tt.total

הפונק' השלישית סוכמת את מס' ההזמנות בכל מדינה:

create function f\_getSumPerCountry ()

returns table

as

return

select country, summationPerCountry=count(\*)

from Shipping\_Times

group by country

הפונק' הרביעית מאחדת את טבלאות הסכימה לפי מדינה עם טבלת הערים הפופולריות (ע"מ שנוכל לרוץ עליה לולאה) ומחשבת את היחס שעליו נריץ את בדיקת התנאי:

create function f\_joinTopCityWithSumPerCountry()

returns table

as

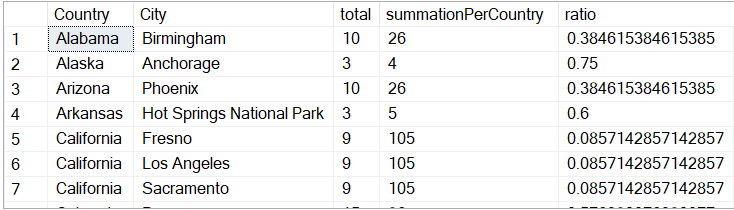
return

select f1.Country,f1.City,f1.total,f2.summationPerCountry,

ratio=cast(cast(f1.total as float)/cast(f2.summationPerCountry as float) as float)

from dbo.f\_getTopCities() as f1 join f\_getSumPerCountry() as f2 on f1.Country=f2.country

group by f1.Country,f1.City,f1.total,f2.summationPerCountry

(יש ערכים נוספים בפלט האמיתי)

עכשיו ניצור SP משני שנועד להכשיר את הטבלה tempTableע"י הוספת עמודת ההחלטה:

create procedure creatingTable

as

create table tempTable (

country varchar(50),

city varchar(50),

total int,

summationPerCountry int,

ratio float

)

insert into tempTable

select \*

from f\_joinTopCityWithSumPerCountry()

alter table tempTable add possibleLocation varchar(3)

נזמן את ה-SP המשני כך:

execute creatingTable

לבסוף, ה- SP הראשי מזמן את הפונקציות ונעזר בהן על מנת לענות על השאלה הסופית כך:

create procedure sp\_getTopCityPerCountry

as

declare c1 cursor for

(

select Country, ratio, possibleLocation

from tempTable

)

declare

@country varchar(20),

@ratio float,

@possibleLocation varchar(3)

open c1

fetch next from c1 into @country,@ratio,@possibleLocation

while @@FETCH\_STATUS=0

begin

if @ratio>0.5

update tempTable

set possibleLocation='Yes'

where country=@country

else

update tempTable

set possibleLocation='No'

where country=@country

fetch next from c1 into @country,@ratio,@possibleLocation

end

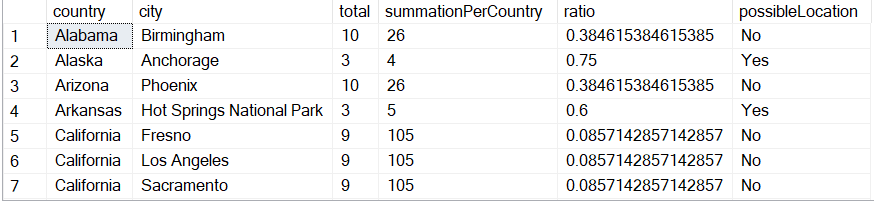
close c1

deallocate c1

select\*

from tempTable

זימון ה-SP יתבצע כך:

execute sp\_getTopCityPerCountry

(יש נתונים נוספים בפלט האמיתי)

לאחר מכן נמחוק את טבלת ה-tempTable כך:

drop table tempTable

## שילוב מערכתי של מספר כלים 10%

המטרה שלנו בחלק זה היא לבצע פעילות שיווקית: בכל סוף חודש אנו רוצים לשלוף באופן רנדומלי 5 אנשים רשומים שביצעו הזמנות. לאנשים הללו ניתן עלות משלוח חינמית ולאלו מהם שמחיר ההזמנה שלהם היה גדול ממוצע מחירי ההזמנות של 5 האנשים, נוסיף פריט אחד נוסף (שזהה לפריט שהם הזמינו) בהזמנה שלהם על חשבוננו. לדעתנו, הגרלה כזאת בסוף חודש עשויה לגרום לאנשים להירשם לאתר ולבצע הזמנות נוספות, ואולי אף לנסות להעלות את מחיר ההזמנה שלהם.

ראשית בנינו viewשמגריל 5 אנשים באופן רנדומלי. View מתאים למשימה כי זו פעולה חוזרת. כמו כן הוספנו לטבלה שדות שרלוונטים לנו להמשך:

create view v\_getWinnersDeatils

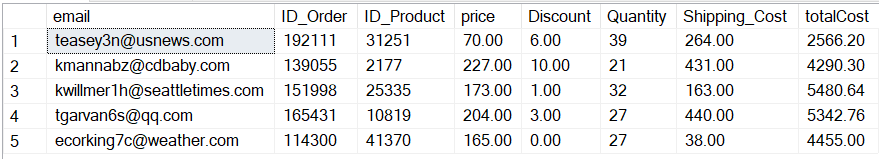
as

SELECT TOP 5 r.email,o.ID\_Order,p.ID\_Product, p.price, o.Discount, c.Quantity, o.Shipping\_Cost,totalCost=price\*Quantity\*(1-discount/100)

FROM Registered as r join orders as o on r.Email=o.Email join containments c on o.ID\_Order=c.ID\_Order

join products as p on c.ID\_Product=p.ID\_Product

ORDER BY NEWID()

(הפלט רנדומלי)

שנית, יצרנו פונק' שמחשבת את ממוצע העלות הכוללת של 5 האנשים וזאת על מנת שרק חלק מהאנשים יקבלו את ההטבה של מתנת מוצר נוסף על חשבוננו:

create function f\_getAvgOfView()

returns smallmoney

as begin

declare @avg smallmoney

select @avg=avg(totalCost)

from v\_getWinnersDeatils

group by totalCost

return @avg

end

לאחר מכן, בנינו SP שבו התייחסו גם ל-view וגם לפונק' שבנינו. ה-SP רץ בלולאה על כל אחד מפרטי 5 האנשים שהוגרלו, ומעדכן להם עלות משלוח 0 (הטבה משותפת לכולם).

כמו כן, עדכנו הזמנות שמחירן הכולל מעל הממוצע כך שיקבלו מוצר נוסף במתנה. SP מתאים למשימה כי זו פעולה חוזרת ומכיוון שאין לנו צורך בערך מוחזר אלא בעדכון, ולכן פונק' לא מתאימה פה. כמו כן רצינו לשלב כלים רבים בגוף אחד, והפונק' לא התאימה לדעתנו.

create procedure sp\_updateDetailsForWinners

as

declare c2 cursor for

select v.Email, v.Quantity, v.Shipping\_Cost,v.totalCost

from v\_getWinnersDeatils as v

declare @email varchar(50),@quantity int,@shipping\_cost smallmoney, @totalCost smallmoney

open c2 fetch next from c2 into @email,@quantity,@shipping\_cost,@totalCost

while @@FETCH\_STATUS=0

begin

update orders

set Shipping\_Cost=0

where Shipping\_Cost=@shipping\_cost

if @totalCost>dbo.f\_getAvgOfView()

update Containments

set quantity=quantity+1

where Quantity=@quantity

fetch next from c2 into @email,@quantity,@shipping\_cost,@totalcost

end

close c2

deallocate c2

הרצת ה-SP יתבצע כך:

execute sp\_updateDetailsForWinners

## דו"ח המושתת על שאילתה מקוננת מורכבת 10%

הדו"ח שלנו מתעמק בנתונים שמתפלגים לפי קטגוריות מוצרי החנות המקוונת. אנו רוצים להשוות בין נתוני המכירות שנוגעים לכל קטגוריה במהלך השנה הנוכחית לעומת השנה הרווחית ביותר בטווח של 5 השנים האחרונות (טווח ריאלי מבחינה עסקית). מכיוון שהתעניינו בנתונים סטטיסטיים הדו"ח מכיל את השדות הבאים: רמת ההתעניינות בכל קטגוריה (כמות חיפושים), מס' ההזמנות שבוצעו בכל קטגוריה, סך ההכנסות לפי קטגוריה, ממוצע ההכנסה לפי קטגוריה בהזמנה בודדת ודירוג הכוכבים הממוצע. בנוסף לכל אלה, צירפנו שדות עם נתונים מהשנה הרווחית ביותר ב-5 השנים האחרונות ע"מ לבצע השוואה פשוטה ונוחה כיוון שכל הנתונים מסודרים באותה טבלה. השדות הנוספים הם: מס' ההזמנות שבוצעו השנת השיא בכל קטגוריה, סה"כ ההכנסות מכל קטגוריה באותה שנה וממוצע ההכנסה לפי קטגוריה בהזמנה בודדת.

הפונק' הראשונה מחזירה את הנתונים שנוגעים לשנה הנוכחית:

CREATE Function infoOnCategories()

returns table

as return

select a1.Category\_Product,a2.ExpressingInterest,a1.numberOfOrders,a1.totalIncom,a1.avgPrice,

a3.avgStars

from (select p.Category\_Product,numberOfOrders=count(\*),totalIncom=sum(p.Price\*c.Quantity),

avgPrice= sum(p.Price\*c.Quantity)/count(\*)

from Orders as o join Containments as c on o.ID\_Order=c.ID\_Order

join Products as p on c.ID\_Product=p.ID\_Product

where YEAR(dt\_order)=YEAR(getdate())

group by p.Category\_Product) as a1

left join (select p.Category\_Product,ExpressingInterest=count(\*)

from Findings as f join Products as p on f.ID\_Product=p.ID\_Product join

Searches as s on s.IP\_Search=f.IP\_Search

where YEAR(s.DT\_Search)=YEAR(getdate())

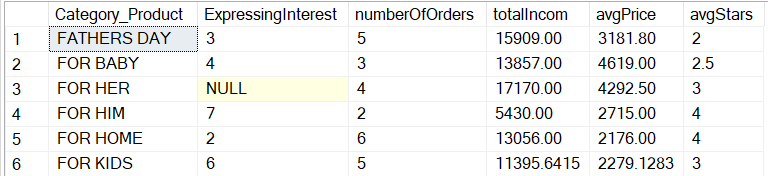
group by p.Category\_Product) as a2 on a1.Category\_Product=a2.Category\_Product

left join (select p.Category\_Product,avgStars=cast((cast(sum(r.Stars)as real)/cast(count(\*)as real))as real)

from Reviews as r join Products as p on p.ID\_Product=r.ID\_Product

where YEAR(r.DT\_Review)=YEAR(getdate())

group by p.Category\_Product) as a3 on a1.Category\_Product=a3.Category\_Product



(יש נתונים נוספים בפלט האמיתי)

הפונק' השנייה מציגה את סך ההכנסות בכל אחת מ-5 השנים האחרונות:

CREATE Function yearsAndIncomes()

returns table

as return

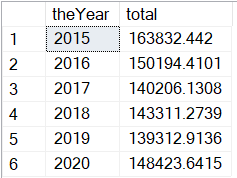
select theYear=YEAR(o.dt\_order),total=sum(p.Price\*c.Quantity)

from Orders as o join Containments as c on o.ID\_Order=c.ID\_Order

join Products as p on c.ID\_Product=p.ID\_Product

where YEAR(getdate())- YEAR(o.dt\_order)<=5

group by YEAR(o.dt\_order)



הפונק' השלישית מחזירה את הנתונים שנוגעים לשנת השיא מבין 5 השנים האחרונות:

CREATE Function infoBestYear()

returns table

as return

select p.Category\_Product,numberOfOrders\_bestYear=count(\*),

totalIncom\_bestYear=sum(p.Price\*c.Quantity),avgPrice\_bestYear= sum(p.Price\*c.Quantity)/count(\*)

from Products as P join Containments as c on p.ID\_Product=c.ID\_Product join Orders as o

on c.ID\_Order=o.ID\_Order

where year(o.dt\_order)=(select dbo.yearsAndIncomes.theYear

from dbo.yearsAndIncomes ()

where dbo.yearsAndIncomes.total=(select MAX(tempT.total)

from dbo.yearsAndIncomes()as tempT))

group by p.Category\_Product



(יש נתונים נוספים בפלט האמיתי)

בחרנו בכלי הפונק' לסייע לנו בכתיבת הקוד כיוון שרצינו לפשט את ביצוע ה-view שיפורט בהמשך. בנוסף, הפעולות שהפונקציות יבצעו לא ישתנו בעתיד הנראה לעין אלא יחזרו על עצמן שוב ושוב.

לבסוף, איחדנו את שני הפלטים בעזרת view בטבלה אחת שיוצרת את הדו"ח הסופי שלנו. בחרנו להשתמש ב-view מכיוון שהפעולה המורכבת שמפורטת פה תתבצע לכל הפחות בכל סוף שנה מחדש ורצינו שלמנהלים יהיה נח לבצע את המשימה. כמובן שמומלץ להריץ את ה-view מס' פעמים טרם סוף שנת העבודה הנוכחית כדי לבחון את סטטוס המכירות והנתונים הרלוונטים הנוספים מראש, כדי לוודא מה הוא קצב המכירות בהתאם ליעד ואל מול חודשים זהים בשנים שעברו, לבצע פעולות מנע במקרה של אי עמידה במשימות ועוד.

create view v\_businessReport

as

select dbo1.Category\_Product,dbo1.ExpressingInterest,dbo1.numberOfOrders,dbo1.totalIncom,dbo1.avgPrice,dbo1.avgStars,

dbo2.numberOfOrders\_bestYear,dbo2.totalIncom\_bestYear,dbo2.avgPrice\_bestYear

from dbo.infoOnCategories() as dbo1 left join infoBestYear() as dbo2 on dbo1.Category\_Product=dbo2.Category\_Product

