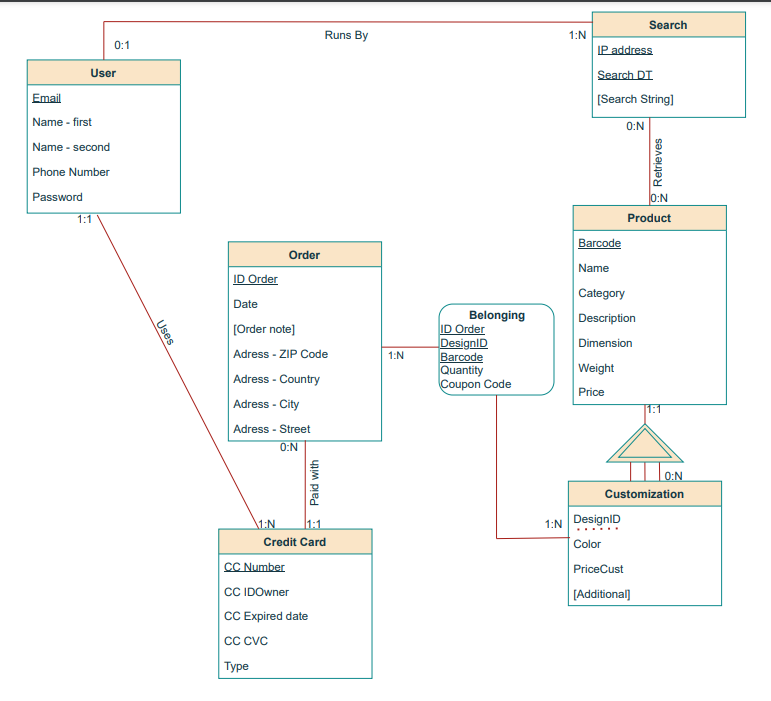
|  |
| --- |
| אתר |
| https://kustomkitgymequipment.com/ |

ERD:



מודל טבלאי:

* **USERS** (Email, Name - first, Name - second, Phone Number, Password)
* **BELONGING** (ID Order (ORDERS), {Design ID, Barcode} (CUSTOMIZATIONS), Quantity, Coupon Code)
* **PRODUCTS** (Barcode, Name, Category, Description, Dimension, Weight, Price)
* **ORDERS** (ID Order, Date, Address - ZIP Code, Address - Country, Address - City, Address- Street, CC Number (CREDITCARDS))
* **ORDERNOTES** (ID Order (ORDERS), Order Note)
* **CREDITCARDS** (CC Number, CC IDOwner, CC Expired date, CC CVC, Type, Email (USERS))
* **SEARCHES** (IP address, Search DT, Email (USERS))
* **SEARCHSTRINGS** ({IP address ,Search DT} (SEARCHES), Search Strings)
* **CUSTOMIZATIONS** (DesignID, Barcode (PRODUCTS), Color, PriceCust)
* **ADDITIONAL** ({DesignID, Barcode} (CUSTOMIZATION), Additional)
* **RETRIVING** (Barcode (PRODUCTS), {IP address, Search DT} (SEARCH))

## מטלה 1 (25%) – שאילתות

שתי שאילתות SELECT ללא קינון

שאילתה 1-

מה הקטגוריה של המוצרים שנמכרה הכי הרבה פעמים בשנת 2023, ומה סך המכירות של קטגוריה זו?

מטרה עסקית: להבין מה הקטגוריה הכי פופולרית בקרב הלקוחות בשנה האחרונה על מנת להבין את עדיפות הלקוחות.

SELECT TOP 1 P.Category ,total = COUNT(\*)

FROM PRODUCTS as P JOIN BELONGING as B on P.Barcode=B.Barcode

JOIN ORDERS as O on O.[ID Order] = B.[ID Order]

WHERE YEAR(O.Date)=2023

GROUP BY P.Category

ORDER BY total DESC

הפלט: רשומה אחת



שאילתה 2-

אילו מוצרים שמחירם קטן מ- 500 ₪, החברה מכרה אותם בסכום גבוה מ- 2500 ₪?

מטרה עסקית: מוצרים שמחירם זול יחסית, ונקנים בכמויות גדולות- משתלמים לחברה כלכלית.

SELECT B.Barcode, [Total Price] = SUM ((P.Price + C.PriceCust)\*B.Quantity)

FROM ORDERS as O JOIN BELONGING as B on O.[ID Order] = B.[ID Order] JOIN CUSTOMIZATIONS as C on C.DesignID=B.DesignID AND C.Barcode=B.Barcode

JOIN PRODUCTS as P on P.Barcode=B.Barcode

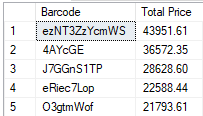
WHERE P.Price < 500

GROUP BY B.Barcode

HAVING SUM (P.Price + C.PriceCust) > 2500

ORDER BY [Total Price] DESC

הפלט: 5 רשומות



שתי שאילתות SELECT מקוננות

שאילתה 1-

השאילתה מחזירה את ממוצע המחיר להזמנה.  
השאילתה המקוננת מחזירה טבלה, המציגה את הסכום של כל הזמנה בודדת המבוצעת באתר.

המטרה העסקית: החברה תהיה מודעת לסכום הזמנה ממוצעת לערך וכך תוכל להעריך את הכנסותיה.

SELECT AVG([Total Price]) AS Average

FROM (

SELECT O.[ID Order],[Total Price] = SUM((P.Price + C.PriceCust)\*B.Quantity)

FROM ORDERS AS O

JOIN BELONGING AS B ON O.[ID Order]= B.[ID Order]

JOIN CUSTOMIZATIONS AS C ON C.DesignID=B.DesignID AND C.Barcode=B.Barcode

JOIN PRODUCTS AS P ON P.Barcode=C.Barcode

GROUP BY O.[ID Order]

)

AS TotalOrderPrice

הפלט: רשומה אחת



שאילתה 2-

השאילתה מחזירה מהו השיעור היחסי של תרומת כל מדינה לרווח הכולל של האתר.  
השאילתה המקוננת מחזירה סקלר שהוא סך ההכנסות

מטרה עסקית: לדעת את רווחיות האתר מהמדינות השונות ולפי כך לדעת איפה לפרסם יותר או פחות.

SELECT O.[Address - Country], [Country Total] = SUM((B.Quantity\*(P.Price+C.PriceCust))),

Precent = SUM((B.Quantity\*(P.Price+C.PriceCust))/ T.Total)

FROM ORDERS AS O

JOIN BELONGING AS B ON O.[ID Order] = B.[ID Order]

JOIN CUSTOMIZATIONS AS C ON C.DesignID = B.DesignID AND C.Barcode = B.Barcode

JOIN PRODUCTS AS P ON C.Barcode = P.Barcode

CROSS JOIN (

SELECT Total = SUM(B.Quantity\*(P.Price+C.PriceCust))

FROM ORDERS AS O

JOIN BELONGING AS B ON O.[ID Order] = B.[ID Order]

JOIN CUSTOMIZATIONS AS C ON C.DesignID = B.DesignID AND C.Barcode = B.Barcode

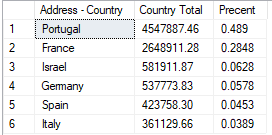
JOIN PRODUCTS AS P ON C.Barcode = P.Barcode

) AS T

GROUP BY T.Total, O.[Address - Country]

ORDER BY Precent DESC

הפלט:



שתי שאילתות מקוננות תוך שימוש במרכיבים נוספים

שאילתה 1-

השאילה מעדכנת את מחיר המוצרים הכי פופולריים (נקנתה מהם הכמות הגדולה ביותר) ובהתאם מחיר כל יחידת מוצר מועלה ב 10%. השאילתה המקוננת מוצאת את 5 המוצרים המבוקשים ומסדרת אותם מהמוצר המבוקש ביותר למוצר הכי פחות מבוקש.

מטרה עסקית: להעלות את מחיר המוצרים המבוקשים ביותר על מנת להעלות את מכירות החברה ורווחיה (היצע וביקוש).

UPDATE PRODUCTS

SET PRICE =Price \* 1.1

WHERE Barcode IN (

SELECT Top 5 Barcode

FROM (

SELECT TOP 5 P.Barcode, P.Price, SUM(CAST(B.Quantity AS INT)) AS Total\_Quantity

FROM BELONGING AS B

JOIN CUSTOMIZATIONS AS C ON C.DesignID = B.DesignID AND C.Barcode = B.Barcode

JOIN PRODUCTS AS P ON C.Barcode = P.Barcode

GROUP BY P.Barcode, P.Price

ORDER BY Total\_Quantity DESC

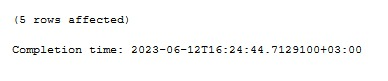
) AS Top\_Products

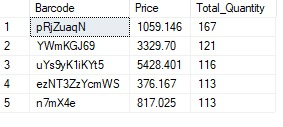
))

הפלט: 5 המוצרים המבוקשים ביותר (לפני העדכון):



לאחר הפעלת השאילתה UPDATE (המחיר ליחידה של כל מוצר עלה ב-10%):





שאילתה 2-

השאילתה מחזירה את כל הלקוחות אשר ביצעו רכישה בשנת 2022 ומאז לא ביצעו רכישות באתר.

מטרה עסקית: שימור לקוחות. החברה מעוניינת לקבל מידע מי הם הלקוחות אשר לא חזרו לרכוש פעם נוספת באתר משנת 2022, ולשלוח להם מסרון עם שאלון האם מרוצים מהמוצרים ומדוע לא נכנסו לרכוש באתר פעם נוספת.

SELECT U.Email, U.[Name - first], U.[Name - second]

FROM USERS AS U JOIN CREDITCARDS AS C ON U.Email = C.Email

JOIN ORDERS AS O ON C.[CC Number]=O.[CC Number]

WHERE YEAR (O.Date) < 2022

GROUP BY U.Email, U.[Name - first], U.[Name - second]

EXCEPT

SELECT U.Email, U.[Name - first], U.[Name - second]

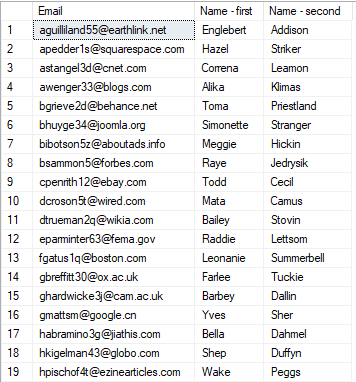
FROM USERS AS U JOIN CREDITCARDS AS C ON U.Email = C.Email

JOIN ORDERS AS O ON C.[CC Number]=O.[CC Number]

WHERE YEAR (O.Date) >=2022

GROUP BY U.Email, U.[Name - first], U.[Name - second]

הפלט: 43 רשומות מתוכן נציג 19



## מטלה 2 (25%) – יישומי כלים מתקדמים

View

השאילתה מחזירה את הרווח היומי לכל מדינה מהרווח הגבוה ביותר לנמוך ביותר, על מנת שלא נצטרך בכל יום מחדש לרשום את השאילתה, נשתמש ב-VIEW.

מטרה עסקית: החברה רוצה להיות בבקרה על המכירות היומיות בכל מדינה ולקבל אינדיקציה באילו מדינות המכירות גבוהות יותר.

CREATE VIEW V\_RevenuePerDay AS

SELECT O.[Address - Country], Revenue = SUM((P.Price + C.PriceCust)\*B.Quantity), [Day]=Day(O.[Date])

FROM CUSTOMIZATIONS AS C JOIN PRODUCTS AS P ON C.Barcode=P.Barcode

JOIN BELONGING AS B ON B.DesignID=C.DesignID AND B.Barcode=C.Barcode

JOIN ORDERS AS O ON O.[ID Order]=B.[ID Order]

GROUP BY O.[Address - Country], Day(O.[Date])

מימוש ה- VIEW:

SELECT R.[Address - Country], R.Revenue

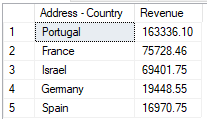
FROM V\_RevenuePerDay AS R

WHERE Day=1

GROUP BY [Address - Country], Revenue

ORDER BY Revenue DESC

הפלט: 5 רשומות



פונקציות (Functions)

שאילתה 1-

הפונקציה מקבלת מזהה של הזמנה ומחזירה את סכום ההזמנה הכולל. פונקציה זו מחזירה סקלר.

מטרה עסקית: מאפשר שליפת נתונים מהירה של סכום הזמנה מכל הזמנה ספציפית שנדרש לבדוק.

DROP FUNCTION PriceForOrder

CREATE FUNCTION PriceForOrder (@INPUT1 Varchar(20))

RETURNS INT

AS BEGIN

DECLARE @OUTPUT\_Price INT

SELECT @OUTPUT\_Price = B.Quantity\*(P.Price+C.PriceCust)

FROM ORDERS AS O

JOIN BELONGING AS B ON O.[ID Order] = B.[ID Order]

JOIN CUSTOMIZATIONS AS C ON C.DesignID = B.DesignID AND   
 C.Barcode = B.Barcode

JOIN PRODUCTS AS P ON C.Barcode = P.Barcode

WHERE O.[ID order] = @INPUT1

RETURN @OUTPUT\_Price

END

קריאה לפונקציה:

SELECT Price = dbo.PriceForOrder('0mHmsp093')

הפלט: רשומה אחת



שאילתה 2-

הפונקציה מחזירה טבלה בה מוצג עבור הלקוח אותו מקבלת כקלט, את כל ההזמנות שביצע ותאריכן.

מטרה עסקית: החברה תוכל לקבל סטטוס הזמנות ותאריכן עבור כל לקוח. החברה תוכל להשתמש בפונקציה זו במקרה הצורך למשל כאשר לקוח מבצע מספר הזמנות ולא זוכר את מספר ההזמנה, נוכל לבדוק אותה לפי התאריך בו ביצע

DROP FUNCTION OrdersByPerson

CREATE FUNCTION OrdersByPerson (@Email Varchar(50))

RETURNS TABLE

AS RETURN

SELECT U.Email, O.Date, O.[ID Order]

FROM ORDERS AS O JOIN CREDITCARDS AS C ON O.[CC Number]=C.[CC Number]

JOIN USERS AS U ON U.Email=C.Email

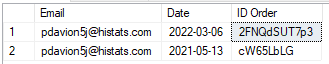
WHERE U.Email= @Email

קריאה לפונקציה:

SELECT \*

FROM dbo.OrdersByPerson('pdavion5j@histats.com')

הפלט: 2 רשומות



Trigger

מכיוון שרצינו לדעת מיד אחרי סיום ההזמנה מה סכום ההזמנה הכולל, הפעלנו טריגר פשוט שלאחר כל הכנסה של נתוני הזמנה הוא יעדכן את עמודת Total Amount בטבלת ORDERS. לשם ביצוע הטריגר נעזרנו בפונקציה שכתבנו קודם שמחשבת את סכום ההזמנה הכולל להזמנה מסוימת .

מטרה עסקית: נרצה לדעת מיד אחרי סיום הזמנה מה סכום ההזמנה הכולל כדי להיות בבקרה על מכירות החברה.

ראשית נוסיף עמודה Total Amount ב- ORDERS ונעדכן את כל ההזמנות שכבר בוצעו עם מחיר ההזמנה הכולל:

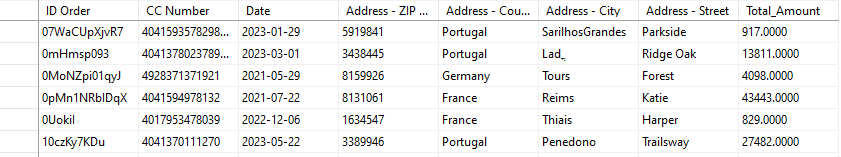
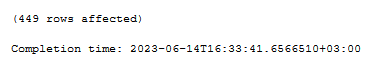
ALTER TABLE ORDERS

ADD Total\_Amount MONEY NULL

UPDATE ORDERS

SET Total\_Amount = dbo.PriceForOrder(ORDERS.[ID Order])

WHERE [ID Order] IN (SELECT DISTINCT [ID Order] FROM ORDERS)



כעת נוסיף את הטריגר שכאשר תיווצר הזמנה חדשה באופן אוטומטי יחשוב עבורה סכום ההזמנה (בתוכו מופעלת פונקציה שכתבנו קודם בעבודה שמחשבת את סכום ההזמנה הכולל):

CREATE TRIGGER Orders\_Amount

ON BELONGING

FOR INSERT

AS

UPDATE ORDERS

SET Total\_Amount = (dbo.PriceForOrder(ORDERS.[ID Order]))

WHERE [ID Order] IN (SELECT DISTINCT [ID Order] FROM INSERTED)

כעת נבצע הכנסה של הנתונים הרלוונטיים:

INSERT INTO CUSTOMIZATIONS VALUES('10382','8cncfsYAi','yellow',62.0000)

INSERT INTO ORDERS VALUES ('07cfgjpghvF7','4017950952321350','2023-01-29','5919841','Israel','Azor','tlalim',null)

INSERT INTO BELONGING VALUES('07cfgjpghvF7','10382','8cncfsYAi',7,'T2kfa7uJ1jg'

זוהי הטבלה המעודכנת:  
   


פרוצדורה שמורה ((Stored Procedure פשוטה

ביצענו פרוצדורה להחלפת סיסמא ללקוח - הלקוח יזין את האימייל שלו ואת הסיסמא החדשה, הפרוצדורה תבדוק האם הסיסמא החדשה שונה מהסיסמא הנוכחית, תתבצע החלפה בין הסיסמאות בבסיס הנתונים ותודפס הודעה שהתבצעה החלפה. אם הסיסמא החדשה זהה לסיסמא הנוכחית תודפס הודעה כי לא ניתן להחליף את הסיסמא.

מטרה עסקית: נוכל להפעיל את הפרוצדורה ללקוחות שהסיסמא שלהם לא שונתה זמן רב ולחייב אותם לשנות אותה.

CREATE PROCEDURE sp\_changePassword

@email VARCHAR(50),

@newPassword VARCHAR(20)

AS

BEGIN

DECLARE @currentPassword VARCHAR(20)

SELECT @currentPassword = [Password]

FROM USERS

WHERE Email = @email

IF @newPassword <> @currentPassword

BEGIN

UPDATE USERS

SET [Password] = @newPassword

WHERE Email = @email

PRINT 'Password changed successfully'

END

ELSE

BEGIN

PRINT 'The new password cannot be the same as the current password'

END

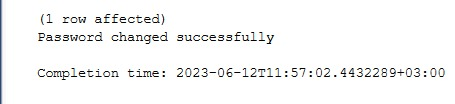
END

קריאה לפרוצדורה:

execute sp\_changePassword 'aguilliland55@earthlink.net','grydhtrkNhr'

מצב של הלוקח לפני שינוי סיסמא 

לאחר הפעלת הפרוצדורה:





## מטלה 3 (20%) – כלים להצגת נתונים

דו"ח עסקי

בחרנו להציג את דו"ח הנתונים העסקיים עבור צוות השיווק של החברה.  
מחלקת שיווק הינה חלק בלתי נפרד מהאתר שלנו, כיוון שהמכירות שלנו מתבססות רק על האתר, חשוב שמחלקת השיווק תיחשף לנתוני המכירות באתר.  
בעזרת הדו"ח, תוכל המחלקה להפיק מסקנות ולדעת היכן להתמקד על מנת להגדיל את המכירות באתר.

הנתונים שהדו"ח מציג:

1. סכום כלל המכירות בכל מדינה לפי שנה
2. אחוז מכירה של כל קטגוריה מסך כל הקטגוריות באתר לפי כל מדינה
3. אחוז מכירה של כל קטגוריה מסך כל הקטגוריות באתר לפי תאריכים
4. סכום המכירה הכולל עבור כל קטגוריה לפי כל מדינה
5. אחוז השימוש בכל סוג כרטיס אשראי
6. סך רווחי החברה לפי תאריכים

את הנתונים ניתן למיין לפי מדינה ותאריכים

הVIEW שכתבנו עבור יצירת הדו"ח העסקי:

DROP VIEW V\_1

CREATE VIEW V\_1

AS

SELECT Country = O.[Address - Country], [Year] = Year(O.Date), P.Category,[Total Orders] = COUNT (O.[ID Order]),

[Total Quantity Per Category] = SUM (CAST(B.Quantity AS Int)) ,[Total Revenue Per Category] = SUM (B.Quantity\*(P.Price + CU.PriceCust))

FROM

PRODUCTS AS P JOIN CUSTOMIZATIONS AS CU ON P.Barcode=CU.Barcode

JOIN BELONGING AS B ON B.Barcode=CU.Barcode AND B.DesignID=CU.DesignID

JOIN ORDERS AS O ON O.[ID Order]=B.[ID Order]

JOIN ORDERNOTES AS ORN ON ORN.[ID Order] = O.[ID Order]

JOIN CREDITCARDS AS CC ON CC.[CC Number]=O.[CC Number]

GROUP BY P.Category, O.[Address - Country], Year(O.Date)

SELECT \* FROM V\_1

DROP VIEW V\_2

CREATE VIEW V\_2

AS

SELECT TotalPaidBy = Count (Distinct O.[ID Order]), cc.Type

FROM

ORDERS AS O JOIN CREDITCARDS AS CC ON O.[CC Number] = CC.[CC Number]

GROUP BY cc.Type

SELECT \* FROM V\_2

--פר לקוח

DROP VIEW V\_3

CREATE VIEW V\_3

AS

SELECT U.Email, O.[Date], O.[Address - Country], CC.Type, P.Category, B.Quantity ,TotalPriceProduct =( P.Price + CU.PriceCust), [TotalExpenses] = SUM (B.Quantity\*(P.Price + CU.PriceCust))

FROM

PRODUCTS AS P JOIN CUSTOMIZATIONS AS CU ON P.Barcode=CU.Barcode

JOIN BELONGING AS B ON B.Barcode=CU.Barcode AND B.DesignID=CU.DesignID

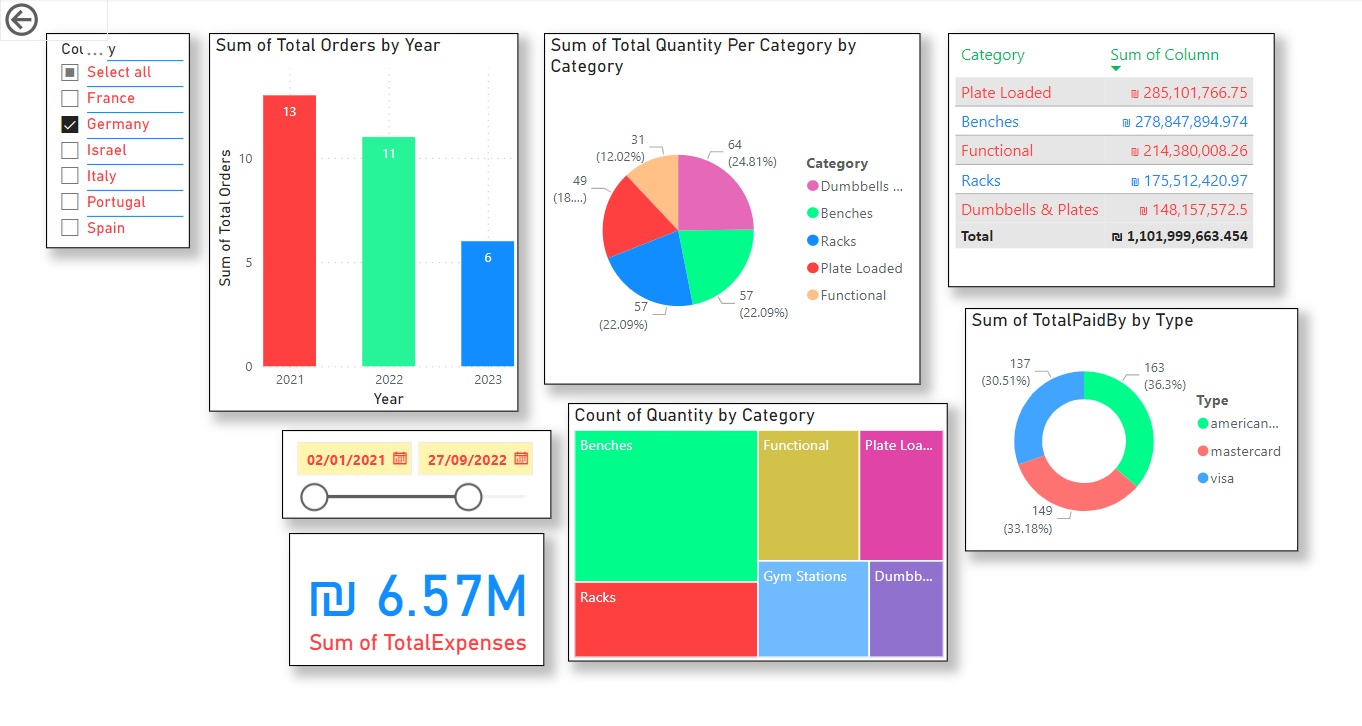
JOIN ORDERS AS O ON O.[ID Order]=B.[ID Order]

JOIN ORDERNOTES AS ORN ON ORN.[ID Order] = O.[ID Order]

JOIN CREDITCARDS AS CC ON CC.[CC Number]=O.[CC Number]

JOIN USERS AS U ON U.Email=CC.Email

GROUP BY U.Email, O.[Date], O.[Address - Country], CC.Type, P.Category, B.Quantity,P.Price, CU.PriceCust

SELECT \* FROM V\_3

לוח מחוונים

לוח מחוונים מיועד להנהלה הבכירה בחברה, ונדרש בכדי להציג עבורם תמונת מצב כוללת על המכירות באתר ופעילותו.  
המידע בלו מלמד את הדרג הניהולי של החברה על הנעשה בה ומסייע לקבלת החלטות מושכלות מבחינה תכנונית – בין אם בקווי בייצור (ביקוש למוצרים - מה לייצר יותר ומה פחות, ביקוש לפי מדינות, תקופות וכו') ובין אם אסטרטגיה שיווקית (באיזה מדינות יש לקדם את החברה, אילו מוצרים כדאי לקדם). בנוסף, המידע יכול לתרום באסטרטגיה ניהולית (אם מוצר מסוים מבוקש מאוד, אולי כדאי להרחיב את אפשרויות המכירה שלו – בצבעים, במידות נוספות – או למשל, להרחיב את שירות הלקוחות במדינה מובילה במכירות ע"מ לשמר את שביעות רצון הלקוחות ולוודא לקוחות חוזרים, וכדומה)

הנתונים שהלוח מציג:

1. המוצר הנמכר ביותר ובאיזה כמות
2. חמשת המשתמשים שרכשו בסכום הגבוה ביותר מאתר החברה
3. סך רווחי החברה בכל סיום חציון
4. סך הזמנות לכל רבעון
5. סך הזמנות
6. סך רווחים

כל אחד מן הנתונים ניתן למיין לפי מדינה, סוג מוצר, וטווח תאריכים

הVIEW שכתבנו עבור יצירת לוח המחוונים:

CREATE VIEW V\_4 AS

SELECT O.[Date], [Year] = Year (O.[Date]) ,O.[ID Order], U.Email ,P.[Name] ,CU.DesignID, P.Barcode ,B.Quantity ,Total = (P.Price + CU.PriceCust) , O.[Address - Country] ,

TotalRevenue = B.Quantity \* (P.Price + CU.PriceCust)

FROM PRODUCTS AS P JOIN CUSTOMIZATIONS AS CU ON P.Barcode=CU.Barcode

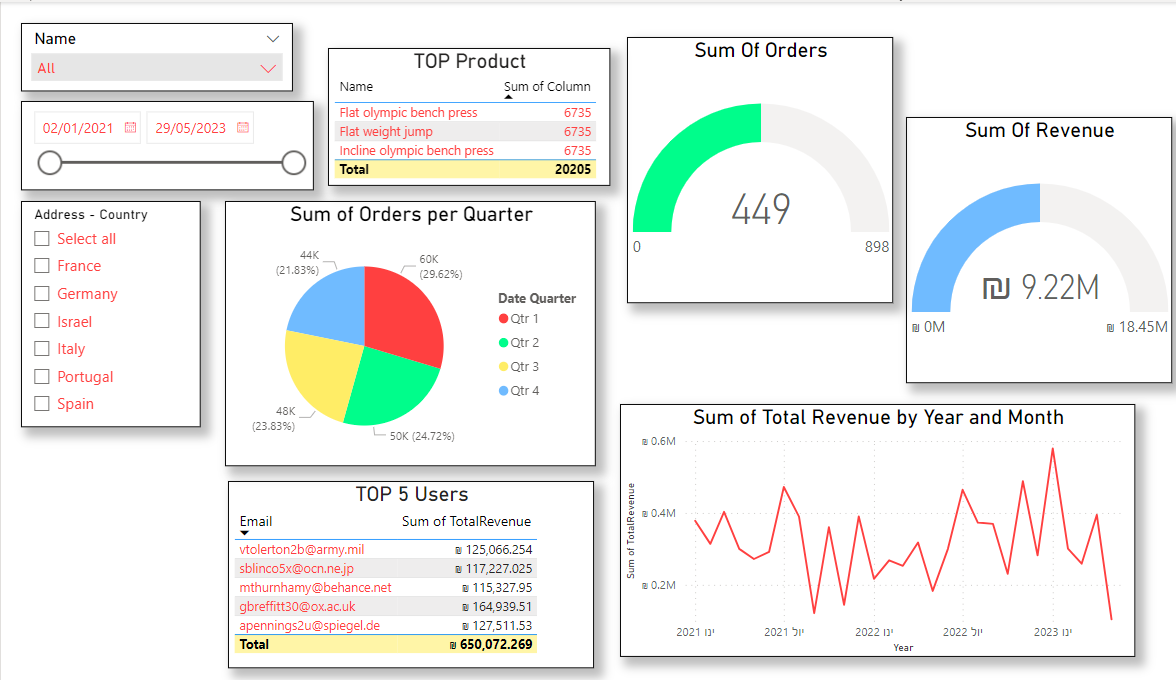
JOIN BELONGING AS B ON B.Barcode=CU.Barcode AND B.DesignID=CU.DesignID

JOIN ORDERS AS O ON O.[ID Order]=B.[ID Order]

JOIN CREDITCARDS AS CC ON CC.[CC Number]=O.[CC Number]

JOIN USERS AS U ON U.Email=CC.Email

GROUP BY O.[Date], O.[ID Order], U.Email ,P.[Name] ,CU.DesignID, P.Barcode ,B.Quantity, P.Price, CU.PriceCust, O.[Address - Country]



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

שאילתות עסקיות המשלבות Window Functions

שאילתה 1-

שאילתה מחזירה טבלה בה מוצג עבור כל שנה את הרווח השנתי של החברה, וגם של שנה הקודמת, מהי עליית/ירידת הרווח באחוזים ובאיזה שנה הייתה העלייה הגבוהה ביותר בחברה.

מטרה עסקית: נרצה למצוא שנים רווחיות על מנת להבין כיצד פעלנו שונה ולשפר את המצב החברה כדי למקסם את הרווחה. בשנים בהן הייתה ירידה, נרצה להבין מה היו הגורמים ומה כדאי לשנות כדי שהמצב ישתפר ויחזור לרגיל.

השאילתה מכילה את הפונקציות- RANK() וLAG():

SELECT [Year], TotalRevenue, LastYearRevenue, LastYearGrowth,

YearRank = RANK () OVER (ORDER BY LastYearGrowth DESC)

FROM (

SELECT \*,

LastYearGrowth = (ROUND((TotalRevenue)/ (LastYearRevenue) ,2)-1 )

FROM

(SELECT \*,

LastYearRevenue = ROUND( LAG(TotalRevenue , 1) OVER (ORDER BY [Year]),2)

FROM

(SELECT [Year] = Year(O.Date),

TotalRevenue = ROUND( SUM(B.Quantity\*(P.Price + C.PriceCust)),2)

FROM ORDERS AS O JOIN BELONGING AS B ON O.[ID Order] = B.[ID Order]

JOIN CUSTOMIZATIONS AS C ON C.DesignID = B.DesignID AND C.Barcode = B.Barcode

JOIN PRODUCTS AS P ON P.Barcode = C.Barcode

GROUP BY Year (O.Date)

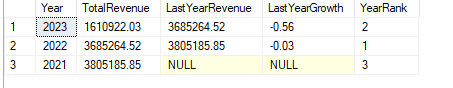
) AS YR

) AS LYG

) AS Final

ORDER BY [Year] DESC

הפלט: 3 רשומות (אחת לכל שנה)



\*לצערנו ניתן לראות שהרווחים באתר שלנו במגמת ירידה ☹

שאילתה 2-

השאילה מחזירה לנו עבור כל קטגוריה, את הלקוח עם הרכישות הכי גבוהות בשנה האחרונה, כולל ממוצעת מכירה פר קטגוריה, מחיר מקסימלי עבור הזמנה של מוצרים מאותה הקטגוריה.

מטרה עסקית: עבור כל קטגוריה נוכל לדעת מי הלקוח שרכש הכי הרבה ובהתאם לשלוח לו פרסומות של מוצרים דומים (של אותה הקטגוריה) על מנת להניע אותו לרכוש עוד.

בשאילתה זאת, השתמשנו בVIEW על מנת להקל על ניתוח הנתונים.

ה VIEW מחזיר לנו טבלה עם פרטי הלקוח, קטגוריות, מחיר ההזמנה, ושנת ההזמנה.

DROP VIEW V\_AllUsersOrders

CREATE VIEW V\_AllUsersOrders

AS

SELECT FullName = U.[Name - first] +' '+ U.[Name - second], P.Category, OrderPrice =SUM ( B.Quantity\*(P.Price+C.PriceCust)), YearOrder = Year(O.[Date])

FROM ORDERS AS O JOIN BELONGING AS B ON O.[ID Order] = B.[ID Order]

JOIN CUSTOMIZATIONS AS C ON C.DesignID=B.DesignID AND C.Barcode= B.Barcode

JOIN PRODUCTS AS P ON P.Barcode = C.Barcode

JOIN CREDITCARDS AS CC ON CC.[CC Number]= O.[CC Number]

JOIN USERS AS U ON U.Email=CC.Email

GROUP BY U.[Name - first],U.[Name - second],Year(O.[Date]) ,P.Category

SELECT \* FROM V\_AllUsersOrders

השאילתה עצמה:

SELECT \*

FROM (

SELECT FullName, Category, OrderPrice,

MaxSalePerCategory = MAX(OrderPrice) OVER(PARTITION BY Category),

AveragePerCategory = AVG(OrderPrice) OVER(PARTITION BY Category),

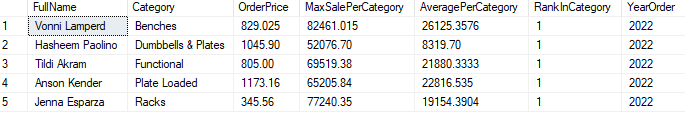
RankInCategory = DENSE\_RANK() OVER(PARTITION BY Category ORDER BY OrderPrice)

,YearOrder

FROM V\_AllUsersOrders

)AS X

WHERE YearOrder= 2022 AND RankInCategory <= 1

פלט: 5 רשומות (אחת פר קטגוריה)

שילוב מערכתי של מספר כלים

בחרנו ליצור שאילתה המורכבת מפונקציה, פרוצדורה שמורה וטריגר. מטרת השאילתה היא להעניק ללקוחות שקנו ביותר מ10,000 ש"ח (סה"כ כל ההזמנות שלהם) הנחה של 1% מסכום הקניות שלהם עד כה. ההנחה תתקבל בדמות נקודות קרדיט שהלקוח יקבל ויוכל לממש אותן בהזמנות הבאות שלו.   
לשם כך יצרנו פונקציה שמחשבת לכל לקוח את הרווח שלו (סוכמת את כל ההזמנות שלו). כמו כן, הפעלנו פרוצדורה שמעדכנת את נקודות הזכות לכל הלקוחות שקנו ביותר מ- 10,000 ₪. הלקוחות מקבלים 1% מסך הרווח שלהם (שימוש בפונקציה) שיישמרו להם בתור שדה חדש שנקרא נקודות זכות.  
על מנת להיות בבקרה על כמות הכסף שהחברה חייבת ללקוחות (לטובת בניית תקציב וחישוב מאזנים) ייצרנו טבלה חדשה בשם REFUNDSאשר שומרת את ההחזרים שהחברה חייבת ללקוחות (נקודות זכות= החזר). לשם הכנסת נתונים לטבלה זו, הפעלנו טריגר שבכל עדכון של טבלת USERS הטבלה מתעדכנת בהתאם לנקודות הזכות שניתנו בפרוצדורה שבנינו.

מטרה עסקית: מתן נקודות זכות לקנייה עתידית ימשוך לקוחות לבצע הזמנות חדשות.

הפונקציה:

DROP FUNCTION getProfit

CREATE FUNCTION getProfit(@email varchar(50))

RETURNS INT

AS BEGIN

DECLARE @OUTPUT\_Profit INT

SELECT @OUTPUT\_Profit = SUM(dbo.PriceForOrder(O.[ID Order]))

FROM USERS AS U JOIN CREDITCARDS AS C ON U.Email=C.Email JOIN ORDERS AS O ON O.[CC Number]=C.[CC Number]

WHERE U.Email=@email

RETURN @OUTPUT\_Profit

END

פלט הפונקציה:

SELECT Profit = dbo.getProfit('aapplebeec@buzzfeed.com')



הוספת שדה נקודות זכות לטבלת USERS:

ALTER TABLE USERS ADD Credit\_points INT NULL

הפרוצדורה השמורה:

CREATE PROCEDURE sp\_getCredit\_points

AS

BEGIN

UPDATE USERS

SET Credit\_points=dbo.getProfit(Email) \* 0.01

SELECT Email, dbo.getProfit(Email) \* 0.01 AS Credit\_points

FROM USERS

WHERE dbo.getProfit(Email) > 10000

GROUP BY Email, dbo.getProfit(Email) \* 0.02

ORDER BY dbo.getProfit(Email) \* 0.02 DESC

END

DROP PROCEDURE sp\_getCredit\_points

פלט הפרוצדורה השמורה:

execute sp\_getCredit\_points

A screenshot of a computer

Description automatically generated with medium confidence

יצירת טבלת REFUNDS:

CREATE TABLE REFUNDS (

Email Varchar(50) not null,

RefundAmount Money ,

[Date] Date not null

Primary Key(Email)

)

DROP TABLE REFUNDS

הטריגר:

CREATE TRIGGER UpdateRefunds

ON USERS

FOR INSERT,UPDATE,DELETE

AS BEGIN

INSERT INTO REFUNDS

SELECT DISTINCT U.Email,U.Credit\_points,GETDATE()

FROM DELETED cross join INSERTED as U

WHERE U.Credit\_points IS NOT NULL

END

העדכון בטבלת REFUNDS:

A screenshot of a computer

Description automatically generated with medium confidence

שליפה מהירה של סכום ההחזרים (החוב) שהחברה חייבת ללקוחות:

SELECT SUM(RefundAmount) As Total\_Depth

FROM REFUNDS



דו"ח המושתת על שימוש בפסקת WITH מורכבת

ביצענו שאילתה מורכבת ובה נקבל טבלה התציג עבור כל מוצר כמה פעמים הוא נמכר, מה כמות המכירות של המוצר, מהי ההכנסה הממוצעת למוצר (מיוצר עם התאמות שונות), מה היחס בין החיפושים להזמנות המוצר ומהו היחס ההפוך. בנוסף, הטבלה תציג את הרווח ממכירת המוצר ואת התרומה שלו המוצר בשימור לקוחות. (הזמנות/ כמה לקוחות הזמינו אותו)

מטרה עסקית: ידע על תרומת המוצר לחברה יעזור לקבל החלטות עסקיות לגביו. לדוגמא, העלאת מחיר המוצר עקב ביקושו.

ALTER TABLE PRODUCTS DROP COLUMN CostProduct

ALTER TABLE PRODUCTS ADD CostProduct real

UPDATE PRODUCTS SET CostProduct = Price\*0.62 --...כמה עולה המוצר

WITH

ListProducts AS (

SELECT ProductBarcode = P.Barcode, TotalRevenue = SUM(B.Quantity \*(P.Price + C.PriceCust)),

AveragePrice = AVG(P.Price + C.PriceCust),

TotalBenefit = SUM(B.Quantity \* (P.Price + C.PriceCust)) - SUM(P.CostProduct \* B.Quantity), Contribution = SUM(B.Quantity \* (P.Price + C.PriceCust)) / COUNT(DISTINCT U.Email)

FROM ORDERS AS O JOIN BELONGING AS B ON O.[ID Order] = B.[ID Order]

JOIN CUSTOMIZATIONS AS C ON C.DesignID=B.DesignID AND C.Barcode= B.Barcode

JOIN PRODUCTS AS P ON P.Barcode = C.Barcode

JOIN CREDITCARDS AS CC ON CC.[CC Number]= O.[CC Number]

JOIN USERS AS U ON U.Email=CC.Email

GROUP BY P.Barcode

),

ListOfOrders AS (

SELECT ProductBarcode = P.Barcode, TotalOrders = COUNT(DISTINCT O.[ID Order]), [Conversion Rate] = Cast(COUNT(DISTINCT O.[ID Order]) as real) / COUNT(DISTINCT R.[Search DT])

FROM ORDERS AS O JOIN BELONGING AS B ON O.[ID Order] = B.[ID Order]

JOIN CUSTOMIZATIONS AS C ON C.DesignID=B.DesignID AND C.Barcode= B.Barcode

JOIN PRODUCTS AS P ON P.Barcode = C.Barcode

JOIN RETRIVING AS R ON R.Barcode = P.Barcode

GROUP BY P.Barcode

),

ListSearches AS (

SELECT R.Barcode, TotalSearches = COUNT(\*)

FROM RETRIVING as R

GROUP BY R.Barcode

),

RatioOrdersSearches AS (

SELECT LO.ProductBarcode, [Search Order Ratio] = CAST(LS.TotalSearches as FLOAT)/LO.TotalOrders

FROM ListOfOrders AS LO

JOIN ListSearches AS LS ON LO.ProductBarcode = LS.Barcode

)

,

TypeToPay AS(

SELECT B.Barcode, O.[ID Order], CC.Type

FROM ORDERS AS O

JOIN BELONGING AS B ON O.[ID Order] = B.[ID Order]

JOIN CREDITCARDS AS CC ON CC.[CC Number] = O.[CC Number]

GROUP BY B.Barcode, O.[ID Order], CC.Type

)

SELECT TOP 10 LP.ProductBarcode, LO.TotalOrders, [Total Sales] = round(LP.TotalRevenue, 2), [Average Price] = round(LP.AveragePrice, 2), [Search Order Ratio] = round(ros.[Search Order Ratio], 4), LO.[Conversion Rate], Profit = round(LP.TotalBenefit, 2), Contribution = round(LP.Contribution, 2)

FROM ListProducts as LP

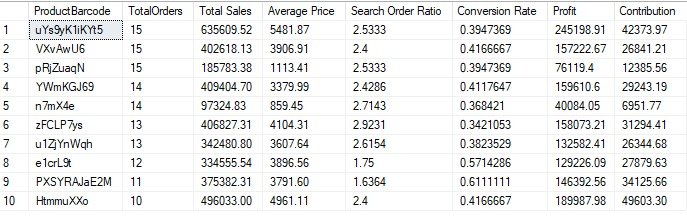
JOIN ListOfOrders as LO ON LP.ProductBarcode = LO.ProductBarcode

JOIN ListSearches as LS ON LS.Barcode = LP.ProductBarcode

JOIN RatioOrdersSearches as ROS ON ROS.ProductBarcode = LP.ProductBarcode

GROUP BY LP.ProductBarcode, LO.TotalOrders, LP.TotalRevenue,ROS.[Search Order Ratio], LP.AveragePrice, LO.[Conversion Rate],LP.TotalBenefit, LP.Contribution

ORDER BY LO.TotalOrders DESC, LP.Contribution DESC

פלט: 10 רשומות