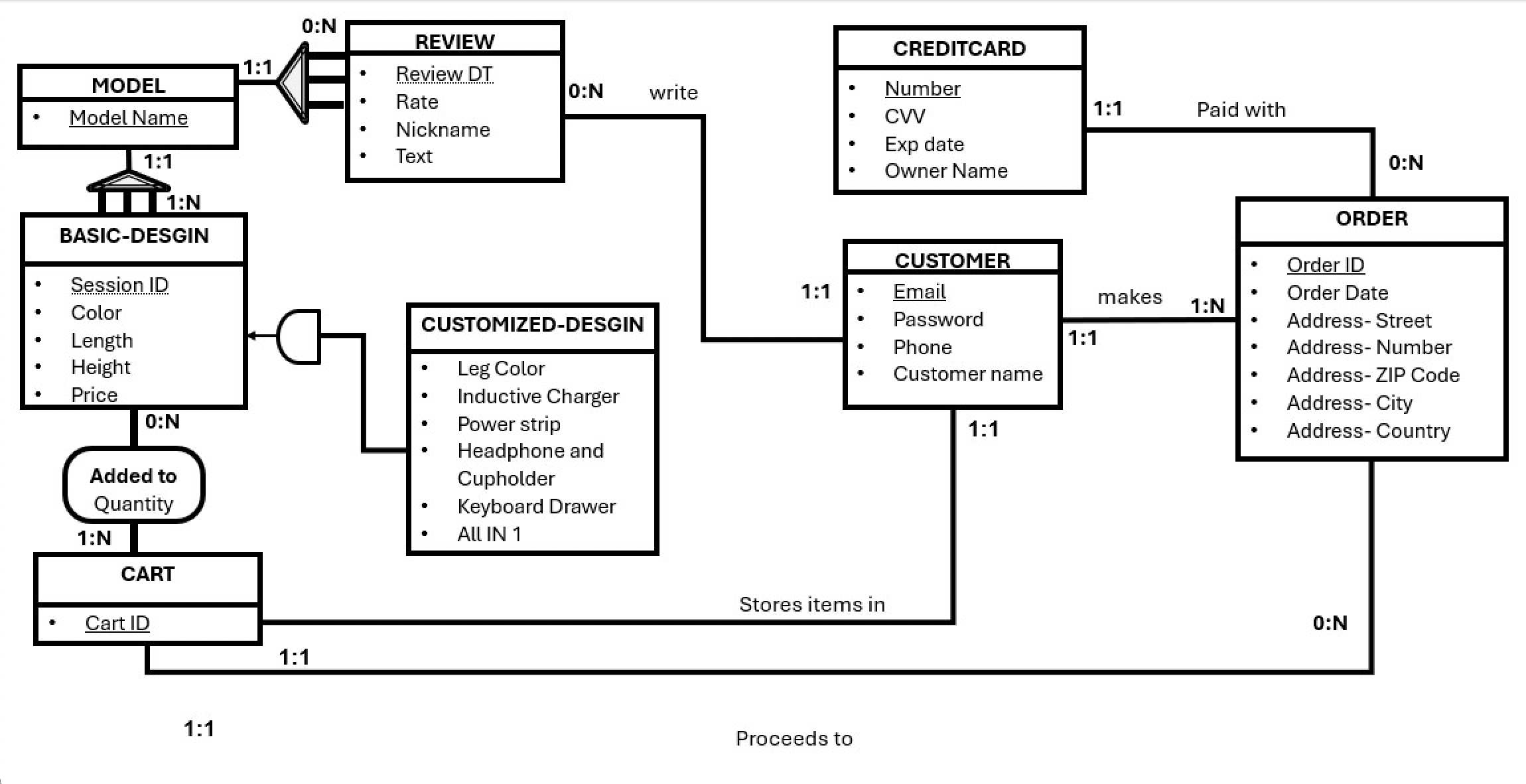
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| מס' קבוצה | אתר | | | תאריך הגשה |
| 10 | <https://gamingart.eu/en/> | | | 26/01/2025 |
| חברי הצוות - מספרי ת.ז | | | | |
| 314711987 | | 206421307 | 319043287 | |

## פרויקט בסיסי נתונים – חלק ג

## פרק ראשון – מטלות חובה

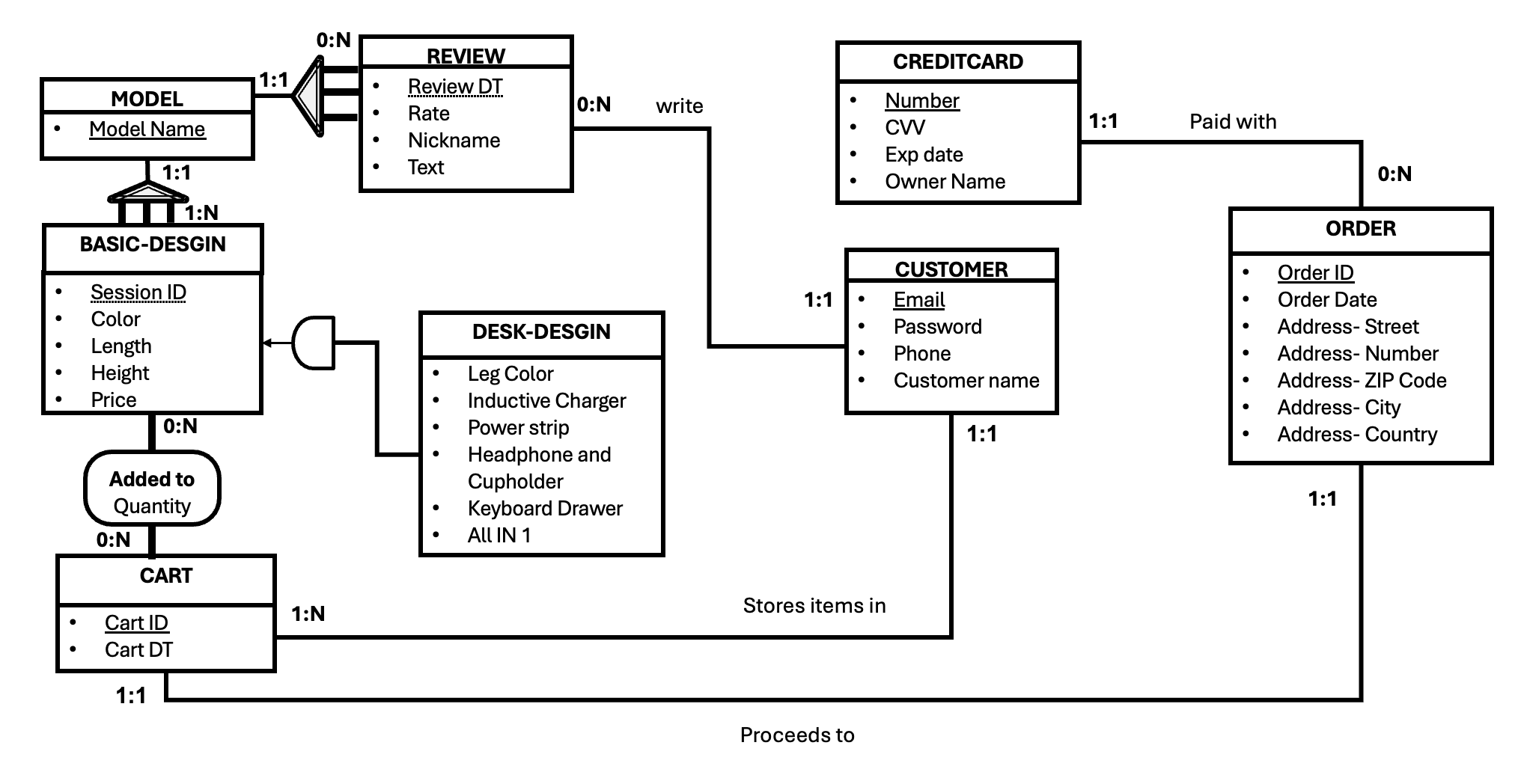
## מטלת חובה מקדימה – תיקון ה-ERD והרחבת היקף בסיס נתונים

ה-ERD הקודם:

הערות שניתנו:

* הקשר בין הטבלאות CUSTOMER ל-ORDER: קשר מיותר, ניכר מהקשר לעגלה
* הקשר בין הטבלאות CART ל-CUSTOMER: בצורה שאתם ממדלים לכל לקוח יכול להיות כמה עגלות. אם ביצעתי שלוש הזמנות בעבר כל עגלה מקושרת לאותו לקוח ושם נשמר מה שהוזמן.
* הקשר בין הטבלאות BASIC-DESIGIN ל-CART: הקרדינליות כאן צריכה להיות הפוכה לא כל עיצוב חייב להיות מוזמן אבל בכל עגלה חייב להיות לפחות מוצר אחד.

ה-ERD המתוקן:



* **Credit Cards**(Number, CVV, Exp date, Owner Name)
* **Customers**(Email, Password, phone, Customer Name)
* **Orders** (Order id, Order Date, Address - Street, Address -  Number, Address - ZIP Code, Address -  City, Address - Country, Credit Card (Credit Cards), Cart ID(Carts))
* **Models** (Model Name)
* **Reviews** (Model Name (Models), DT, Rate, Nickname, Text, Email (Customers))
* **Basic Designs** (Model Name (Models), SessionID, Length, Height, Color, Price)
* **Desk Designs** ({Model Name ,SessionID}(Basic Designs), Leg Color, Inductive Charger, Power strip, Headphone and Cupholder, Keyboard Drawer, All IN 1)
* **Cart** (Cart ID, Cart DT, Email (Customers))
* **Added to** ({Model Name, SessionID}(Basic Designs), Cart ID (Carts), Quantity)

## מטלה 1 – שאילתות

# שתי שאילתות SELECT ללא קינון

**שאילתה 1:**

השאילתה מציגה את עשרת הלקוחות המובילים מבחינת הכנסות שנוצרו מרכישת מוצרים בשנת 2024, כולל כתובות האימייל שלהם, כשהמיון מתבצע בסדר יורד לפי סך ההכנסות.

SELECT TOP 10

CU.[Customer Name] AS [Customer],

CU.Email AS [Email],

SUM(AT.Quantity \* BD.Price) AS [TotalRevenue]

FROM AddedTo AT

INNER JOIN Carts CT

ON AT.CartID = CT.CartID

INNER JOIN Customers CU

ON CT.Email = CU.Email

INNER JOIN BasicDesigns BD

ON AT.ModelName = BD.ModelName AND AT.SessionID = BD.SessionID

INNER JOIN Orders O

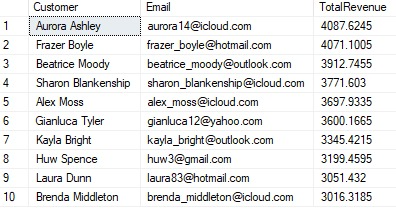
ON CT.CartID = O.CartID

WHERE O.OrderDate >= DATEADD(YEAR, -1, GETDATE())

GROUP BY CU.[Customer Name], CU.Email

ORDER BY SUM(AT.Quantity \* BD.Price) DESC

פלט השאילתה:



**הסבר עסקי:**

השאילתה מספקת מידע על הלקוחות שמייצרים את ההכנסות הגבוהות ביותר, מה שמאפשר לעסק להתמקד בשימורם, להציע להם שירותים מותאמים אישית או הטבות נוספות. בנוסף, הנתונים תורמים לתכנון קמפיינים שיווקיים ממוקדים ללקוחות בעלי פרופיל דומה, במטרה להרחיב את בסיס הלקוחות הרווחיים.

**תרחיש עסקי:**

העסק יכול ליצור תוכנית נאמנות ללקוחות, שבה ישתמש בשאילתה זו כדי לזהות לקוחות ששווה להשקיע בהם יותר ולהציע להם בונוסים, הנחות או מבצעים בלעדיים, מתוך מטרה להגדיל את רמת המחויבות שלהם לעסק.

**שאילתה 2:**

השאילתה מציגה את צבעי המוצרים הפופולריים ביותר שמחירם מעל 200 דולר, ומציגה את הצבעים עם סך הרכישות הגבוה ביותר, כאשר נמכרו לפחות 20 יחידות מכל צבע.

SELECT

BD.Color AS [Most Wanted Color],

COUNT(\*) AS [Total Orders],

SUM(AT.Quantity) AS [Total Model Purchases]

FROM AddedTo AT

INNER JOIN BasicDesigns BD

ON AT.ModelName = BD.ModelName AND AT.SessionID = BD.SessionID

INNER JOIN DeskDesigns DD

ON BD.ModelName = DD.ModelName AND BD.SessionID = DD.SessionID

INNER JOIN Orders O

ON AT.CartID = O.CartID

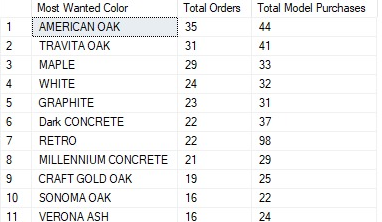
WHERE BD.Price > 200

GROUP BY BD.Color

HAVING SUM(AT.Quantity) >= 20

ORDER BY COUNT(\*) DESC;

פלט השאילתה:



**מטרה עסקית:**

ניתוח צבעי השולחנות הפופולריים מסייע לעסק להתאים את המלאי לדרישות השוק ולמנוע חוסרים או עודפים. בנוסף, המידע מאפשר יצירת קמפיינים שיווקיים ממוקדים, כמו מבצעים על הצבעים המבוקשים ביותר.

**תרחיש עסקי:**

## יצרן הרהיטים מזהה את הצבעים הפופולריים ביותר עבור שולחנות גדולים ומתמקד בהגדלת המלאי של צבעים אלו תוך הפחתה בצבעים פחות מבוקשים. גישה זו מסייעת במניעת חוסרים במלאי לצבעים מבוקשים ובהפחתת עלויות אחסון למוצרים שאינם נמכרים היטב.

# שתי שאילתות SELECT מקוננות

**שאילתה 1:**השאילתה מציגה את 5 הדגמים הנמכרים ביותר, כאשר הדגמים עומדים בסף דירוג ממוצע של מעל 2.5 בביקורת. התוצאות ממוינות לפי הכמות הכוללת בסדר יורד.

SELECT TOP 5

AT.ModelName,

SUM(AT.Quantity) AS [Total Quantity]

FROM AddedTo AT

INNER JOIN Orders O

ON AT.CartID = O.CartID

WHERE AT.ModelName IN (

SELECT R.ModelName

FROM Reviews R

GROUP BY R.ModelName

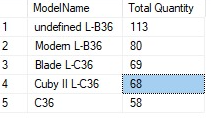
HAVING AVG(R.Rate) > 2.5

)

GROUP BY AT.ModelName

ORDER BY [Total Quantity] DESC;

פלט השאילתה:



**מטרה עסקית:**

השאילתה מזהה את הדגמים הפופולריים ביותר שזכו לדירוג ביקורות חיובי, מה שמאפשר לעסק למקד את המלאי והמכירות במוצרים אלו. כך ניתן לשפר את שביעות רצון הלקוחות ולהגדיל את הרווחיות.

**תרחיש עסקי:**

לדוגמה, יצרן הרהיטים מעוניין לזהות אילו דגמים הם המוצלחים ביותר מבחינת מכירות, תוך שמירה על איכות גבוהה על פי משוב הלקוחות. תובנות אלו מאפשרות לו להגדיל את הייצור והפרסום של מוצרים אלו ולהתמקד בהגדלת הרווחיות תוך שמירה על שביעות רצון הלקוחות.

**שאילתה 2:**

השאילתה מזהה מדינות עם יותר מ-10 הזמנות סך הכל, שבהן מחיר ההזמנה הממוצע גבוה מהממוצע הגלובלי. התוצאות ממוינות לפי מחיר הזמנה ממוצע בסדר יורד.

SELECT

Orders.AddressCountry AS Country,

AVG(BasicDesigns.Price \* Addedto.Quantity) AS [Average Order Price],

COUNT(Orders.OrderID) AS [Total Orders]

FROM Orders

JOIN Addedto ON Orders.CartID = Addedto.CartID

JOIN BasicDesigns ON Addedto.ModelName = BasicDesigns.ModelName

AND Addedto.SessionID = BasicDesigns.SessionID

WHERE Orders.AddressCountry IN (

SELECT Orders.AddressCountry

FROM Orders

GROUP BY Orders.AddressCountry

HAVING COUNT(Orders.OrderID) > 10

)

GROUP BY Orders.AddressCountry

HAVING AVG(BasicDesigns.Price \* Addedto.Quantity) > (

SELECT AVG(BD.Price \* AT.Quantity)

FROM Orders O

JOIN Addedto AT ON O.CartID = AT.CartID

JOIN BasicDesigns BD ON AT.ModelName = BD.ModelName

AND AT.SessionID = BD.SessionID

)

ORDER BY [Average Order Price] DESC;

פלט השאילתה:



**מטרה עסקית:**

השאילתה מסייעת לזהות מדינות עם מכירות משמעותיות ומחיר הזמנה ממוצע גבוה מהממוצע הגלובלי. כך ניתן למקד משאבים שיווקיים במדינות רווחיות ולהגדיל את ההכנסות.

**תרחיש עסקי:**

בהתבסס על הנתונים מהשאילתה, החברה יכולה להשיק קמפיינים לפריטים יוקרתיים במדינות עם מחירי הזמנות ממוצעים גבוהים, וכך להגדיל את הרווחיות שלה.

## שאילתות עסקיות המשלבות Window Functions

**שאילתה 1:**

השאילתה מחשבת את סך ההכנסות על פי רבעונים, ומציגה את שיעור הצמיחה באחוזים ביחס לחודש הקודם. היא מאפשרת לעקוב אחר מגמות הצמיחה או הירידה בהכנסות לאורך זמן בצורה מפורטת.

SELECT

YEAR(O.OrderDate) AS Year,

MONTH(O.OrderDate) AS Month,

NTILE(4) OVER (PARTITION BY YEAR(O.OrderDate) ORDER BY MONTH(O.OrderDate)) AS Quarter, -- Split into quarters

SUM(BD.Price \* A.Quantity) AS TotalRevenue, -- Total revenue for the month

CASE

WHEN LAG(SUM(BD.Price \* A.Quantity)) OVER (PARTITION BY YEAR(O.OrderDate) ORDER BY MONTH(O.OrderDate)) IS NULL THEN '0.00%'

ELSE

CAST(ROUND(

(SUM(BD.Price \* A.Quantity)

/ LAG(SUM(BD.Price \* A.Quantity)) OVER (PARTITION BY YEAR(O.OrderDate) ORDER BY MONTH(O.OrderDate)) - 1) \* 100, 2

) AS VARCHAR) + '%'

END AS MonthlyGrowthPercentage -- Monthly growth as a percentage with a % sign

FROM

Orders O

JOIN AddedTo A ON O.CartID = A.CartID

JOIN BasicDesigns BD ON A.ModelName = BD.ModelName AND A.SessionID = BD.SessionID

GROUP BY YEAR(O.OrderDate), MONTH(O.OrderDate)

ORDER BY Year, Quarter, Month;

פלט השאילתה:



**מטרה עסקית:**

המטרה העסקית של השאילתה היא לעקוב אחר מגמות ההכנסות החודשיות, לזהות שינויים בצמיחה או ירידות בין חודשים סמוכים, ולנתח ביצועים ברמת רבעונים. מידע זה מסייע לעסק לזהות תקופות חזקות או חלשות, להתאים אסטרטגיות שיווקיות ולשפר את תהליכי קבלת ההחלטות העסקיות.

**תרחיש עסקי:**

חברת רהיטים מגלה שבחודש יוני יש ירידה של 15% בהכנסות לעומת חודש מאי, על אף שמאי נמצא באותו רבעון. ניתוח נוסף באמצעות השאילתה מצביע על כך שהירידה נובעת ממכירות נמוכות של דגם מסוים שהיה מבוקש מאוד באפריל ובמאי. בעקבות זאת, החברה מחליטה להשיק מבצע ייחודי על הדגם ביוני ולהקצות תקציב פרסום נוסף, במטרה לשפר את הביצועים החודשיים ולהשלים את היעד הרבעוני.

**שאילתה 2:**

השאילתה מחשבת את ההכנסות הכוללות לכל דגם בכל שנה, מדרגת את הדגמים לפי אחוזונים(Percentile Rank) ביחס לשאר הדגמים באותה שנה, ומסווגת אותם לקטגוריות, לפי מיקומם באחוזון. היא מאפשרת זיהוי של דגמים מובילים או חלשים בכל שנה לפי תרומתם להכנסות.

SELECT

BD.ModelName AS ModelName,

YEAR(O.OrderDate) AS Year,

SUM(BD.Price \* A.Quantity) AS Revenue, -- Total revenue for the model

ROUND(PERCENT\_RANK() OVER (PARTITION BY YEAR(O.OrderDate) ORDER BY SUM(BD.Price \* A.Quantity) ASC), 2) AS PercentileRank, -- Rounded Revenue percentile rank

CASE

WHEN ROUND(PERCENT\_RANK() OVER (PARTITION BY YEAR(O.OrderDate) ORDER BY SUM(BD.Price \* A.Quantity) ASC), 2) <= 0.33 THEN 'Low-End'

WHEN ROUND(PERCENT\_RANK() OVER (PARTITION BY YEAR(O.OrderDate) ORDER BY SUM(BD.Price \* A.Quantity) ASC), 2) <= 0.66 THEN 'Mid-Range'

WHEN ROUND(PERCENT\_RANK() OVER (PARTITION BY YEAR(O.OrderDate) ORDER BY SUM(BD.Price \* A.Quantity) ASC), 2) > 0.95 THEN 'BEST SELLING'

ELSE 'High-End'

END AS RevenueSegment -- Segment based on revenue percentiles

FROM

Orders O

JOIN Carts C ON O.CartID = C.CartID

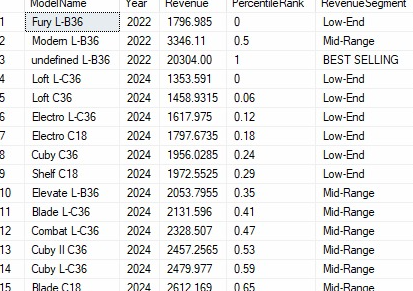
JOIN AddedTo A ON C.CartID = A.CartID

JOIN BasicDesigns BD ON A.ModelName = BD.ModelName AND A.SessionID = BD.SessionID

GROUP BY BD.ModelName, YEAR(O.OrderDate)

ORDER BY Year, PercentileRank;

פלט השאילתה:



**מטרה עסקית:**

המטרה העסקית היא לנתח את הביצועים הפיננסיים של כל דגם ביחס לשאר, כדי למקד שיווק בדגמים מובילים, לשפר דגמים חלשים, ולמקסם רווחים מקטגוריות ביניים.

**תרחיש עסקי:**

## חברת רהיטים מגלה באמצעות השאילתה שדגם מסוים סווג כ-"BEST SELLING" בשנה האחרונה, אך מספר דגמים אחרים נמצאים בקטגוריית "Low-End". החברה מחליטה למקד את מאמצי השיווק בדגם המצליח כדי לשמור על המומנטום, ולהשיק שדרוגים או מבצעים לדגמים החלשים כדי לשפר את ביצועיהם או לפנות מלאי.

## שאילתה מקוננת תוך שימוש ב-CTE (פסקת WITH)

השאילתה מזהה דגמים פופולריים עם מספר הזמנות גבוה מהממוצע או החציון ומנתחת את שילובי הצבעים שלהם (צבע משטח ורגליים). בנוסף, היא מספקת נתונים על כמות השילובים ומחיריהם הממוצעים, תוך התמקדות בדגמים המבוקשים ביותר.

WITH

ModelOrders AS (

-- Subquery 1: Calculate the total number of orders for each model

SELECT

A.ModelName,

COUNT(DISTINCT O.OrderID) AS TotalOrders

FROM AddedTo A

JOIN Orders O ON A.CartID = O.CartID

GROUP BY A.ModelName

),

MedianAndAverage AS (

-- Subquery 2: Calculate the average and median of TotalOrders using window functions

SELECT

DISTINCT

AVG(TotalOrders) OVER () AS AvgOrders, -- Average over all TotalOrders

PERCENTILE\_CONT(0.75) WITHIN GROUP (ORDER BY TotalOrders) OVER () AS MedianOrders

FROM ModelOrders

),

ValidModels AS (

-- Subquery 3: Filter models that exceed the median or average

SELECT

MO.ModelName

FROM ModelOrders MO

CROSS JOIN MedianAndAverage MA

WHERE MO.TotalOrders > MA.MedianOrders

),

CombinationDetails AS (

-- Subquery 4: Calculate total sales and average price for each combination (Color, LegColor) for valid models

SELECT

A.ModelName,

BD.Color,

DD.LegColor,

COUNT(\*) AS CombinationCount,

AVG(BD.Price) AS AveragePrice -- Calculate average price for the combination

FROM AddedTo A

JOIN BasicDesigns BD ON A.ModelName = BD.ModelName AND A.SessionID = BD.SessionID

JOIN DeskDesigns DD ON BD.ModelName = DD.ModelName AND BD.SessionID = DD.SessionID

WHERE A.ModelName IN (SELECT ModelName FROM ValidModels) -- Include only valid models

GROUP BY A.ModelName, BD.Color, DD.LegColor

)

-- Final query: Retrieve combination details along with average price

SELECT

CD.ModelName,

CD.Color,

CD.LegColor,

CD.CombinationCount,

CD.AveragePrice

FROM CombinationDetails CD

ORDER BY CD.ModelName, CD.CombinationCount DESC;

פלט השאילתה:



**מטרה עסקית:**

המטרה העסקית של השאילתה היא לזהות שילובי צבעים פופולריים של שולחן ורגליים, ובכך לאפשר לחברה לשקול להוסיף את השילובים המבוקשים כמוצרים מוכנים למכירה באתר.

**תרחיש עסקי:**

למשל עבור הדגם Blade L-C36, השילוב של CRAFT GOLD OAK עם רגליים בצבע שחור שמופיע הכי הרבה פעמים החברה יכולה לשקול להוסיף את שילוב זה כמוצר מוכן באתר, ולשפר את חוויית הלקוח עם הצעות מדויקות יותר.

## מטלה 2 – יישומי כלים מתקדמים

# View

השאילתה יוצרת View בשם CustomerBehavior שמרכזת נתונים על התנהגות הלקוחות במטרה לנתח את הרגלי הרכישה שלהם. ה-View מספק מידע כולל על פעילות הלקוחות, כולל הוצאות כוללות, ערך הזמנה ממוצע, הרגלי רכישה, והערכת סיכון נטישה.

CREATE VIEW CustomerBehavior AS

SELECT

CU.Email AS CustomerEmail,

SUM(BD.Price \* AT.Quantity) AS TotalSpent,

AVG(BD.Price \* AT.Quantity) AS AvgOrderValue,

SUM(AT.Quantity) AS TotalQuantity,

COUNT(DISTINCT CT.CartID) AS TotalOrders,

MIN(O.OrderDate) AS FirstPurchaseDate,

MAX(O.OrderDate) AS LastPurchaseDate,

(SELECT TOP 1 BD.Color

FROM AddedTo AT2

JOIN BasicDesigns BD ON AT2.ModelName = BD.ModelName AND AT2.SessionID = BD.SessionID

JOIN Carts CT2 ON AT2.CartID = CT2.CartID

WHERE CT2.Email = CU.Email

GROUP BY BD.Color

ORDER BY COUNT(\*) DESC) AS PreferredColor,

CASE

WHEN DATEDIFF(DAY, MAX(O.OrderDate), GETDATE()) > 60 THEN 'High'

ELSE 'Low'

END AS ChurnRisk -- Indicates the likelihood of churn based on inactivity

FROM Customers CU

JOIN Carts CT ON CU.Email = CT.Email

JOIN AddedTo AT ON CT.CartID = AT.CartID

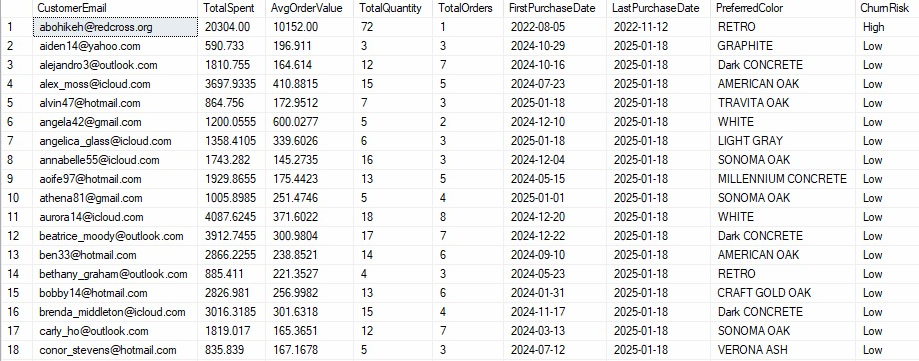
JOIN BasicDesigns BD ON AT.ModelName = BD.ModelName AND AT.SessionID = BD.SessionID

JOIN Orders O ON CT.CartID = O.CartID

GROUP BY CU.Email;

SELECT \* FROM CustomerBehavior

פלט השאילתה:



**מקרים בהם ניתן להשתמש בview-:**

* ניתוח התנהגות לקוחות: מאפשר לעסק להבין את הרגלי הרכישה של הלקוחות, מה הם קונים וכמה הם מוציאים.
* חיזוי נטישה: זיהוי לקוחות בסיכון לנטוש כדי להתערב בזמן באמצעות הצעות או קמפיינים ממוקדים.
* שיפור נאמנות הלקוחות: שימוש במידע כמו צבע מועדף ליצירת קמפיינים מותאמים אישית.
* זיהוי לקוחות רווחיים: פילוח לקוחות בעלי ערך גבוה לצורך שימור והגדלת מכירות.

# פונקציות (Functions)

**פונקציה שמחזירה טבלה:**

הפונקציה מזהה לקוחות שביצעו יותר מרכישה אחת ומחשבת את ממוצע הימים בין הרכישות שלהם. היא מסננת לקוחות עם ממוצע ימים קטן מהערך שניתן כפרמטר, כדי לאתר לקוחות שרוכשים לעיתים קרובות.

CREATE FUNCTION dbo.GetFrequentBuyers (@MaxDaysBetweenPurchases INT)

RETURNS TABLE

AS

RETURN

(

SELECT

CU.Email AS CustomerEmail,

CU.[Customer Name],

COUNT(DISTINCT O.OrderDate) AS TotalPurchaseDays,

MIN(O.OrderDate) AS FirstPurchaseDate,

MAX(O.OrderDate) AS LastPurchaseDate,

CASE

WHEN COUNT(DISTINCT O.OrderDate) <= 1 THEN NULL -- Undefined if only 1 or no purchases

ELSE CAST(

CAST(DATEDIFF(DAY, MIN(O.OrderDate), MAX(O.OrderDate)) AS DECIMAL) /

NULLIF((COUNT(DISTINCT O.OrderDate) - 1), 0)

AS DECIMAL(18, 2)

)

END AS AvgDaysBetweenPurchases

FROM Customers CU

JOIN Carts CT ON CU.Email = CT.Email

JOIN Orders O ON CT.CartID = O.CartID

GROUP BY CU.Email, CU.[Customer Name]

HAVING

COUNT(DISTINCT O.OrderDate) > 1 AND -- Ensure at least two purchases to avoid division by zero

CASE

WHEN COUNT(DISTINCT O.OrderDate) <= 1 THEN NULL

ELSE CAST(

CAST(DATEDIFF(DAY, MIN(O.OrderDate), MAX(O.OrderDate)) AS DECIMAL) /

NULLIF((COUNT(DISTINCT O.OrderDate) - 1), 0)

AS DECIMAL(18, 2)

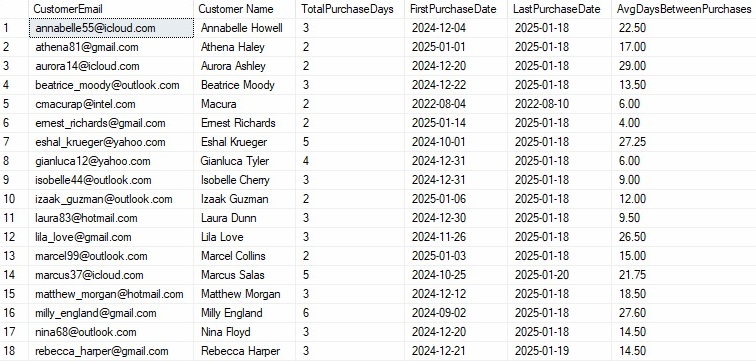
)

END < @MaxDaysBetweenPurchases

);

SELECT \* FROM dbo.GetFrequentBuyers(30);

פלט הפונקציה:



**הצדקה עסקית:**

ההצדקה של הפונקציה היא לזהות לקוחות שרוכשים בתדירות גבוהה על מנת לשמר אותם באמצעות הצעות מותאמות אישית ותוכניות נאמנות. בנוסף, היא מאפשרת לזהות לקוחות שרוכשים בתדירות נמוכה ולפעול להגדלת תדירות הרכישות שלהם באמצעות קמפיינים שיווקיים ואסטרטגיות מיקוד מתאימות.

**פונקציה שמחזירה סקלר:**

הפונקציה dbo.GetCartConversionRate מחשבת את שיעור ההמרה של עגלות שהפכו להזמנות בשנה מסוימת. היא מקבלת כפרמטר את השנה, מחשבת את מספר העגלות הכולל שנוצרו בשנה זו ואת מספר העגלות שהומרו להזמנות, ומחזירה את שיעור ההמרה באחוזים.

CREATE FUNCTION dbo.GetCartConversionRate (@Year INT)

RETURNS DECIMAL(5, 2)

AS

BEGIN

DECLARE @TotalCarts INT;

DECLARE @ConvertedCarts INT;

DECLARE @ConversionRate DECIMAL(5, 2);

-- Count the total carts created in the specified year

SELECT @TotalCarts = COUNT(\*)

FROM Carts

WHERE YEAR(CartDT) = @Year;

-- Count the carts that were converted into orders

SELECT @ConvertedCarts = COUNT(DISTINCT O.CartID)

FROM Carts C

JOIN Orders O ON C.CartID = O.CartID

WHERE YEAR(C.CartDT) = @Year;

-- Calculate the conversion rate

IF @TotalCarts = 0

SET @ConversionRate = 0; -- Avoid divide-by-zero

ELSE

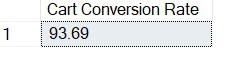
SET @ConversionRate = CAST(@ConvertedCarts AS DECIMAL(5, 2)) \* 100 / @TotalCarts;

RETURN @ConversionRate;

END;

SELECT dbo.GetCartConversionRate(2025) AS [Cart Conversion Rate];

פלט הפונקציה:



**הצדקה עסקית:**

הפונקציה מספקת מדד חשוב המאפשר לעסק לעקוב אחר שיעור ההמרה של עגלות להזמנות בכל שנה נתונה. נתון זה מאפשר:

* מעקב אחר ביצועים עסקיים: זיהוי מגמות בשיעור ההמרה לאורך השנים.
* אופטימיזציה של חוויית הקנייה: אם שיעור ההמרה נמוך, ניתן לשפר את חוויית המשתמש באתר, כמו הצעות מותאמות אישית בזמן תהליך הקנייה.
* תכנון אסטרטגי: הבנת אחוז העגלות שהופכות להזמנות יכולה לעזור בהערכה של יעילות קמפיינים שיווקיים ובהגדלת שיעור ההמרה באמצעות פעולות ממוקדות.

# Trigger

חלק 1: יצירת עמודה חדשה בטבלה Models ועדכון ממוצע דירוגים אוטומטי באמצעות Trigger

ALTER TABLE Models

ADD AverageRate DECIMAL(4, 2) NULL;

GO

--Create trigger

CREATE TRIGGER tr\_UpdateModelsAvgRate

ON Reviews

AFTER INSERT, UPDATE, DELETE

AS

BEGIN

UPDATE M

SET M.AverageRate = (

SELECT AVG(CAST(R.Rate AS DECIMAL(10, 2)))

FROM Reviews R

WHERE R.ModelName = M.ModelName

)

FROM Models M

WHERE M.ModelName IN

(

SELECT ModelName FROM Inserted

UNION

SELECT ModelName FROM Deleted

);

END;

GO

חלק 2: הצעת מבצע ללקוחות שנתנו דירוגים נמוכים:

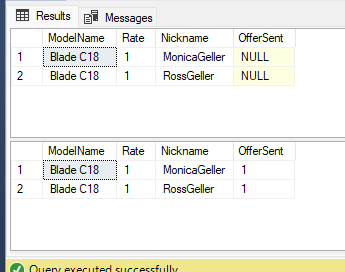
ALTER TABLE Reviews

ADD OfferSent BIT DEFAULT 0;

SELECT ModelName, Rate, Nickname, OfferSent

FROM Reviews

WHERE Rate < 3;



עדכון הערך בעמודה OfferSent ל-1 עבור ביקורות עם דירוג נמוך מ-3, רק עבור דגמים שהממוצע הכללי שלהם (AverageRate) נמוך מ-4.

UPDATE Reviews

SET OfferSent = 1

WHERE Rate < 3

AND ModelName IN (

SELECT ModelName

FROM Models

WHERE AverageRate < 4

);

SELECT ModelName, Rate, Nickname, OfferSent

FROM Reviews

WHERE Rate < 3;

תמונה שמכילה טקסט, צילום מסך, תצוגה, מספר

התיאור נוצר באופן אוטומטי

**מטרה עסקית:**

* מעקב אחר איכות הדגמים: שמירה על ממוצע הדירוגים לכל דגם בטבלת Models מאפשרת מעקב אחרי ביצועי הדגמים לאורך זמן וזיהוי בעיות.
* שיפור חוויית הלקוח: שליחת הצעות ללקוחות שהביעו חוסר שביעות רצון (דירוגים נמוכים) היא כלי לשימור לקוחות ולשיפור מוניטין המותג.
* זיהוי בעיות בדגמים: זיהוי דגמים עם ממוצע דירוגים נמוך (AverageRate < 4) כדי לשפרם או להציע מבצעים ללקוחות שרכשו אותם.

# פרוצדורה שמורה ((Stored Procedure

הפרוצדורה ApplyBonusToHighSpenders מזהה לקוחות שהוציאו סכום מעל סף מסוים (@ThresholdAmount) במהלך ששת החודשים האחרונים ומעניקה להם בונוס בצורת כרטיס מתנה. סכום הבונוס מחושב לפי נוסחה לוגורית שמדרגת את הסכום בהתאם להוצאות הלקוח, עם מינימום של 10 ומקסימום של 35 (אחוזים מהסכום שמעל הסף). הפרוצדורה מציגה את הלקוחות שזכאים לכרטיס המתנה, יחד עם הסכום הכולל שהוציאו וגובה כרטיס המתנה.

CREATE PROCEDURE ApplyBonusToHighSpenders

@ThresholdAmount MONEY -- Input: Minimum total spent to qualify for the bonus

AS

BEGIN

-- Temporary table to store customers eligible for the bonus

CREATE TABLE #EligibleCustomers (

Email VARCHAR(50),

[Customer Name] VARCHAR(100),

TotalSpent MONEY,

GiftCardAmount INT

);

-- Calculate the total amount spent by each customer in the last 6 months and determine the gift card amount

INSERT INTO #EligibleCustomers (Email, [Customer Name], TotalSpent, GiftCardAmount)

SELECT

C.Email,

C.[Customer Name],

SUM(B.Price) AS TotalSpent, -- Calculate total spent

CASE

WHEN SUM(B.Price) > @ThresholdAmount THEN

CAST(

ROUND(

CASE

-- Calculate the percentage using the logarithmic function

WHEN 10 + LOG(1 + (SUM(B.Price) - @ThresholdAmount) / @ThresholdAmount) \* 17 > 35

THEN 35

WHEN 10 + LOG(1 + (SUM(B.Price) - @ThresholdAmount) / @ThresholdAmount) \* 17 < 10

THEN 10

ELSE 10 + LOG(1 + (SUM(B.Price) - @ThresholdAmount) / @ThresholdAmount) \* 17

END \* 10, -- Convert the percentage into a gift card amount

0 -- Round to the nearest whole number

) AS INT

)

ELSE 0

END AS GiftCardAmount

FROM

Carts CA

INNER JOIN Customers C ON CA.Email = C.Email

INNER JOIN Orders O ON CA.CartID = O.CartID -- Include only completed orders

INNER JOIN AddedTo A ON CA.CartID = A.CartID

INNER JOIN BasicDesigns B ON A.ModelName = B.ModelName AND A.SessionID = B.SessionID

WHERE

O.OrderDate >= DATEADD(MONTH, -6, GETDATE()) -- Only include orders from the last 6 months

GROUP BY C.Email, C.[Customer Name];

-- Display the customers eligible for the bonus along with their gift card amount

SELECT

EC.Email AS Customer\_Email,

EC.[Customer Name] AS Customer\_Name,

EC.TotalSpent AS Total\_Spent,

EC.GiftCardAmount AS Gift\_Card\_Amount

FROM #EligibleCustomers EC;

-- Drop the temporary table

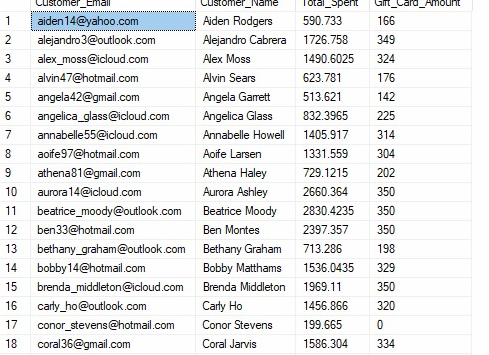
DROP TABLE #EligibleCustomers;

END;

-- Execute the procedure

EXEC ApplyBonusToHighSpenders @ThresholdAmount = 400;

פלט הפרוצדורה:



**המטרה העסקית:**

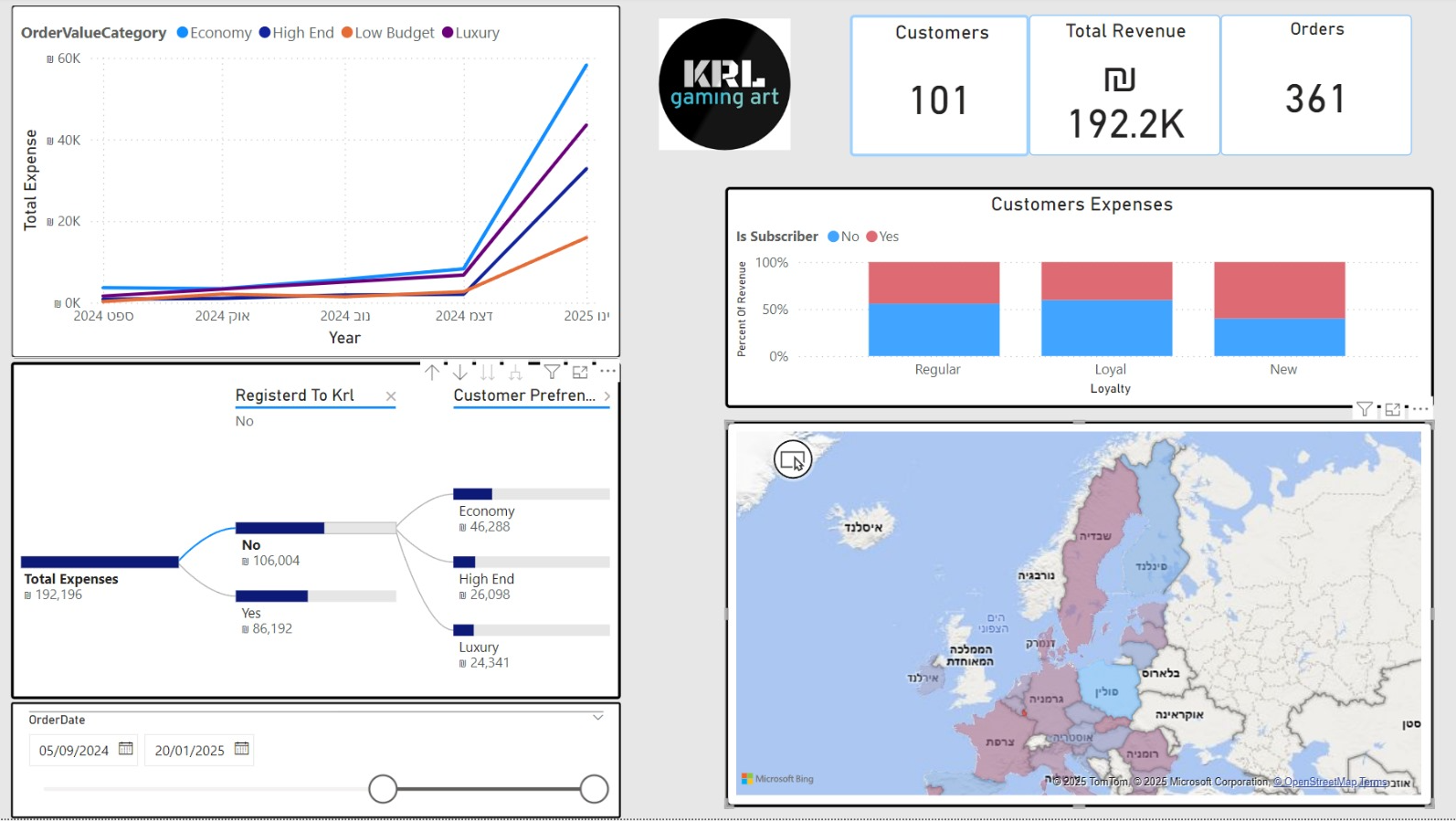
המטרה העסקית היא לתגמל לקוחות שמבצעים רכישות משמעותיות כדי לשמר את נאמנותם ולעודד אותם להמשיך לרכוש. מתן בונוס מותאם אישית מגדיל את שביעות הרצון ומחזק את הקשר עם לקוחות בעלי ערך גבוה.

## מטלה 3 – כלים להצגת נתונים

**דו"ח עסקי -**



**לוח מחוונים -**



## טבלאות סיכום ו-VIEWS שסייעו בהפקת הכלים:

CREATE VIEW PowerBI\_View AS

WITH OrderMetrics AS (

SELECT

o.OrderID,

SUM(bd.Price \* at.Quantity) AS OrderTotalPrice,

CASE

WHEN SUM(bd.Price \* at.Quantity) < 300 THEN 'Low Budget'

WHEN SUM(bd.Price \* at.Quantity) BETWEEN 300 AND 800 THEN 'Economy'

WHEN SUM(bd.Price \* at.Quantity) BETWEEN 800 AND 1100 THEN 'High End'

ELSE 'Luxury'

END AS OrderValueCategory

FROM Orders o

LEFT JOIN Carts cart ON o.CartID = cart.CartID

LEFT JOIN AddedTo at ON cart.CartID = at.CartID

LEFT JOIN BasicDesigns bd ON at.ModelName = bd.ModelName AND at.SessionID = bd.SessionID

GROUP BY o.OrderID

),

CustomerMetrics AS (

SELECT

c.Email AS CustomerEmail,

COUNT(o.OrderID) AS TotalOrders,

AVG(om.OrderTotalPrice) AS AvgOrderPricePerCustomer,

CASE

WHEN COUNT(o.OrderID) < 3 THEN 'New'

WHEN COUNT(o.OrderID) BETWEEN 3 AND 6 THEN 'Regular'

ELSE 'Loyal'

END AS CustomerLoyaltyCategory

FROM Customers c

LEFT JOIN Carts cart ON c.Email = cart.Email

LEFT JOIN Orders o ON cart.CartID = o.CartID

LEFT JOIN OrderMetrics om ON o.OrderID = om.OrderID

GROUP BY c.Email

)

SELECT

-- Order Details

o.OrderID,

o.OrderDate,

o.AddressCity AS City,

o.AddressCountry AS Country,

-- Cart Details

cart.CartID,

cart.CartDT AS CartDateTime,

-- Customer Details

c.Email AS CustomerEmail,

c.[Customer Name] AS CustomerName,

c.Phone AS CustomerPhone,

c.Password AS CustomerPassword, -- Include the password column for subscription check

-- Model Details

m.ModelName,

bd.SessionID,

bd.Color AS DesignColor,

bd.Price AS DesignPrice,

-- Added To Cart Details

at.Quantity,

(bd.Price \* at.Quantity) AS TotalPrice,

-- Review Details

r.Rate AS ReviewRating,

bd.Color AS Color,

-- Aggregated Metrics from Subqueries

om.OrderTotalPrice,

om.OrderValueCategory,

cm.AvgOrderPricePerCustomer,

cm.CustomerLoyaltyCategory,

-- Is Subscriber

CASE

WHEN c.Password IS NOT NULL AND c.Password <> '' THEN 'Yes'

ELSE 'No'

END AS IsSubscriber,

-- Categorical Metric: Price Range of Products

CASE

WHEN bd.Price < 200 THEN 'Economical'

WHEN bd.Price BETWEEN 200 AND 500 THEN 'Mid-Range'

ELSE 'Premium'

END AS PriceRangeCategory,

-- Categorical Metric: Order Season

CASE

WHEN MONTH(o.OrderDate) IN (12, 1, 2) THEN 'Winter'

WHEN MONTH(o.OrderDate) IN (3, 4, 5) THEN 'Spring'

WHEN MONTH(o.OrderDate) IN (6, 7, 8) THEN 'Summer'

ELSE 'Fall'

END AS OrderSeasonCategory,

-- Aggregated Review Metrics

AVG(r.Rate) OVER (PARTITION BY at.ModelName) AS AvgProductReviewRating,

COUNT(r.Rate) OVER (PARTITION BY at.ModelName) AS TotalProductReviews

FROM Orders o

LEFT JOIN Carts cart ON o.CartID = cart.CartID

LEFT JOIN Customers c ON cart.Email = c.Email

LEFT JOIN AddedTo at ON cart.CartID = at.CartID

LEFT JOIN BasicDesigns bd ON at.ModelName = bd.ModelName AND at.SessionID = bd.SessionID

LEFT JOIN Models m ON bd.ModelName = m.ModelName

LEFT JOIN Reviews r ON r.ModelName = bd.ModelName AND r.Email = c.Email

LEFT JOIN OrderMetrics om ON o.OrderID = om.OrderID

LEFT JOIN CustomerMetrics cm ON c.Email = cm.CustomerEmail

WHERE o.OrderID IS NOT NULL; -- Ensure only rows with orders are included

SELECT \*

FROM PowerBI\_View;

## מטלה 4 – אופטימיזציה של שאילתות באמצעות Generative AI

[קישור לשיחה עם הצ’אט](https://chatgpt.com/share/67955caa-91cc-800d-8df0-46c184d8e4d6)

**שאילתה 1-**השאילתה שבחרנו לשפר היא השאילתה המקוננת השנייה בעבודה.

WITH CountryOrderStats AS (

SELECT

Orders.AddressCountry AS Country,

AVG(BasicDesigns.Price \* Addedto.Quantity) AS AvgOrderPrice,

COUNT(Orders.OrderID) AS TotalOrders

FROM Orders

JOIN Addedto ON Orders.CartID = Addedto.CartID

JOIN BasicDesigns ON Addedto.ModelName = BasicDesigns.ModelName

AND Addedto.SessionID = BasicDesigns.SessionID

GROUP BY Orders.AddressCountry

HAVING COUNT(Orders.OrderID) > 10

),

GlobalAvg AS (

SELECT AVG(BD.Price \* AT.Quantity) AS GlobalAvgPrice

FROM Orders O

JOIN Addedto AT ON O.CartID = AT.CartID

JOIN BasicDesigns BD ON AT.ModelName = BD.ModelName

AND AT.SessionID = BD.SessionID

)

SELECT

COS.Country,

COS.AvgOrderPrice AS [Average Order Price],

COS.TotalOrders AS [Total Orders]

FROM CountryOrderStats COS

CROSS JOIN GlobalAvg GA

WHERE COS.AvgOrderPrice > GA.GlobalAvgPrice

ORDER BY COS.AvgOrderPrice DESC;

**שאילתה 2-**שאילתה נוספת ששיפרנו היא שאילתת החלון השנייה בעבודה.

-- Create Indexes for Optimization

CREATE INDEX idx\_orders\_orderdate\_cartid

ON Orders (OrderDate, CartID);

CREATE INDEX idx\_carts\_cartid

ON Carts (CartID);

CREATE INDEX idx\_addedto\_cartid\_modelname\_sessionid\_quantity

ON AddedTo (CartID, ModelName, SessionID, Quantity);

CREATE INDEX idx\_basicdesigns\_modelname\_sessionid\_price

ON BasicDesigns (ModelName, SessionID, Price);

-- Optimized Query

WITH RevenueData AS (

SELECT

BD.ModelName AS ModelName,

YEAR(O.OrderDate) AS Year,

SUM(BD.Price \* A.Quantity) AS Revenue -- Total revenue for the model

FROM

Orders O

JOIN Carts C ON O.CartID = C.CartID

JOIN AddedTo A ON C.CartID = A.CartID

JOIN BasicDesigns BD ON A.ModelName = BD.ModelName AND A.SessionID = BD.SessionID

GROUP BY BD.ModelName, YEAR(O.OrderDate)

),

RankedRevenue AS (

SELECT

ModelName,

Year,

Revenue,

PERCENT\_RANK() OVER (PARTITION BY Year ORDER BY Revenue ASC) AS PercentileRank -- Percentile rank calculation

FROM

RevenueData

)

SELECT

ModelName,

Year,

Revenue,

ROUND(PercentileRank, 2) AS PercentileRank, -- Rounded Revenue percentile rank

CASE

WHEN ROUND(PercentileRank, 2) <= 0.33 THEN 'Low-End'

WHEN ROUND(PercentileRank, 2) <= 0.66 THEN 'Mid-Range'

WHEN ROUND(PercentileRank, 2) > 0.95 THEN 'BEST SELLING'

ELSE 'High-End'

END AS RevenueSegment -- Segment based on revenue percentiles

FROM

RankedRevenue

ORDER BY Year, PercentileRank;