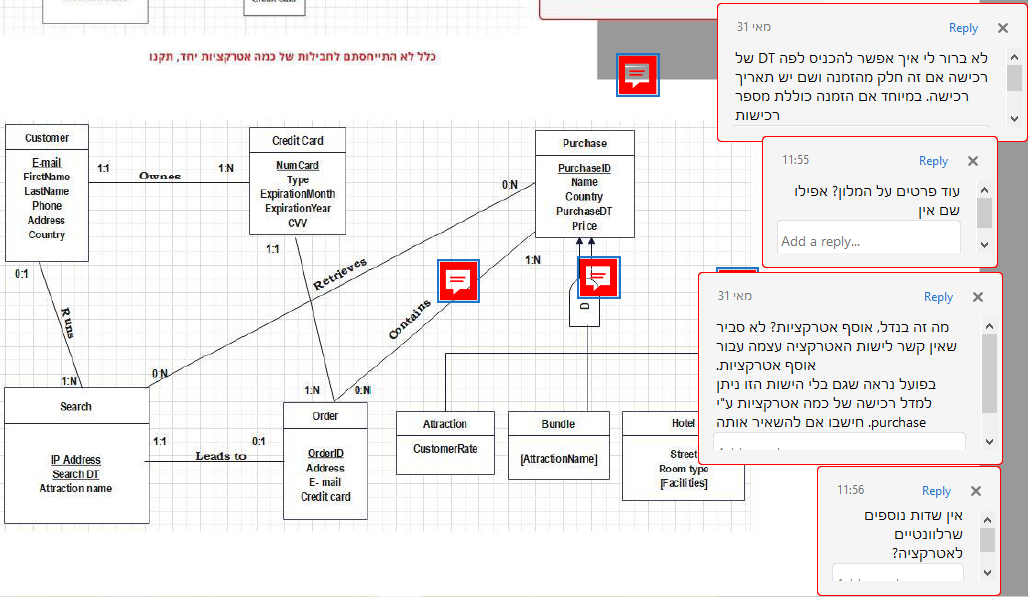
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| מס' קבוצה | אתר | | | תאריך הגשה |
| 18 | <https://www.attractiontickets.com/en> | | | 27/06/2022 |
| חברי הצוות – מספרי ת.ז | | | | |
| 209133222 | | 316468586 | 208634196 | |

## פרויקט בסיסי נתונים

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

## ERD חלק ב':



## תמונה שמכילה טקסט התיאור נוצר באופן אוטומטימודל טבלאי חלק ב':

## ERD מתוקן:

## מודל טבלאי מתוקן:

* Customers (Email, First Name , Last Name, Phone, Address, Country)
* Credit Cards (NumCard, Types, Experation-Year, Experation -Month, CVV, Email(CUSTOMER))
* Orders (OrderID, E-mail, OrderDT, NumCard (Credit Cards), ({IPAddress, SearchDT} (Searches))
* Searches (IP Address, SearchDT, ActivityName, Email(Customers))
* Purchases(PurchaseID, Country, Price)
* Attractions (PurchaseID(Purchases), ActivityName ,CustomerRate, PurchaseDT)
* Bundles (Bundle(ATTRACTION), BundleName)
* Hotels (PurchaseID(Purchases), Street, RoomType, HotelName)
* Facilities (PurchaseID (HOTELS), Facility)
* Includes (OrderID (ORDERS), PurchaseID ((Purchases))
* Retrieves ({IPAddress, SearchDT} (SEARCHES), PurchaseID (Purchases))
* נרמול: לאחר שבדקנו נרמול מסדר 3NF , לא היה צורך בנרמול נוסף של הטבלאות

**הנחות:**

|  |  |  |
| --- | --- | --- |
| **מרכיב** | **הנחות** | **הערות נוספות** |
| Entity: Customer   * Email * Name * Phone * Address * Country | * -Email האתר מגביל לכתובת מייל אחת ללקוח. |  |
| הקשר בין Search לבין Purchase | לא כל חיפוש יוביל לסוג של רכישה אשר קיימת באתר.  מספר חיפושים שונים עשויים להוביל לאותה הרכישה. |  |

|  |  |  |  |
| --- | --- | --- | --- |
| **Table** | **Field** | **Data Type** | **Size (Bytes)** |
| **Customers** | Email | Varchar(50) | 50+2 |
| FirstName | Varchar(10) | 10+2 |
| LastName | Varchar(10) | 10+2 |
| Phone | Varchar(30) | 30+2 |
| Address | Varchar(30) | 30+2 |
| Country | Varchar(30) | 30+2 |
| **Total Record Size (Bytes)** | | **172** |
| Credit Cards | NumCard | Varchar(16) | 16+2 |
| Types | Varchar(30) | 30+2 |
| Experation-Month | TinyInt | 1 |
| Experation-Year | SmallInt | 2 |
| CVV | Varchar(3) | 3+2 |
| Email | Varchar(50) | 50+2 |
| **Total Record Size (Bytes)** | | **108** |
| Purchases | PurchaseID | Integer | 4 |
| Country | Varcher(30) | 30+2 |
| Price | SmallMoney | 4 |
| **Total Record Size (Bytes)** | | **40** |
| Searches | IP Address | Varchar(30) | 30+2 |
| SearchDT | Datetime | 8 |
| ActivityName | Varchar(30) | 30+2 |
| Email | Varchar(50) | 50+2 |
| **Total Record Size (Bytes)** | | **124** |
| Attractions | PurchaseID | Integer | 4 |
| CustomerRate | TinyInt | 1 |
| ActivityName | Varchar(30) | 30+2 |
| PurchaseDT | Datetime | 8 |
| **Total Record Size (Bytes)** | | **45** |
| Hotels | PurchaseID | Integer | 4 |  |
| Street | Varchar(30) | 30+2 |  |
| RoomType | Varchar(10) | 10+2 |  |
| HotelName | Varchar(30) | 30+2 |  |
| **Total Record Size (Bytes)** | | **80** |  |
| Facilities | PurchaseID | Integer | 4 |  |
| Facility | Varchar(30) | 30+2 |  |
| **Total Record Size (Bytes)** | | **36** |  |
| Bundles | Bundles | Varchar(30) | 30+2 |  |
| BundleName | Varchar(30) | 30+2 |  |
| **Total Record Size (Bytes)** | | **64** |  |
| Orders | OrderID | Integer | 4 |  |
| E-mail | Varchar(50) | 50+2 |  |
| NumCard | Varchar(50) | 50+2 |  |
| IPAddress | Varchar(15) | 15+2 |  |
| SearchDT | Datetime | 8 |  |
| OrderDT | Datetime | 8 |  |
| **Total Record Size (Bytes)** | | **141** |  |
| Includes | OrderID | Integer | 4 |  |
| PurchaseID | Integer | 4 |  |
|  | **Total Record Size (Bytes)** | | **8** |  |
| Retrieves | IPAddress | Varchar(30) | 30+2 |  |
| SearchDT | Datetime | 8 |  |
| PurchaseID | Integer | 4 |  |
| **Total Record Size (Bytes)** | | **44** |  |

**הנחות:**

יכולים להתחיל ב0 ולכן הם מוגדרים כואצ'אר.- Cvv, Account

CustomerRATE- הינו מספר מ 1-10.

**תיקון - אילוצי בדיקה (CHECK):**

* **אילוץ על Email**: הוספת אילוץ על השדה Email בטבלה.  
  דרך המייל החברה מזהה את הלקוח באופן חד-ערכי ויוצרת עימו קשר. שדה זה ייחודי עבור כל לקוח ולכן צריך לבדוק את תקינותו.
* **הגדרת טווח לדירוג האטרקציה**– הגדרת טווח בין 1-10.
* **הגדרת טווח רצוי לתוקף כרטיס האשראי**- הגבלת תוקף האשראי מבחינת חודש ותאריך לתקינים.
* **הגדרת מחיר מינימום**- הגבלת המחיר לגבוה מ- 0.

ALTER TABLE Customers ADD CONSTRAINT CK\_EMAIL CHECK (Email LIKE '%@%.%')

ALTER TABLE Purchases ADD CONSTRAINT CK\_Price Check (Price>0)

ALTER TABLE Attractions ADD CONSTRAINT CK\_CustomerRate Check (CustomerRate>0 AND CustomerRate<11)

ALTER TABLE CreditCards ADD CONSTRAINT CK\_Expiration\_month Check (Expiration\_month >0 AND Expiration\_month <13)

ALTER TABLE CreditCards ADD CONSTRAINT CK\_Expiration\_Year Check (Expiration\_Year>Year(getdate())

**טבלאות חיפוש (Lookup Tables):**

**טבלת חיפוש** **ל-RoomType:**הוספנו טבלת חיפוש עבור סוג חדר (RoomType) מתוך הנחה כי לקוח יכול לבחור באחד מבין ארבעת סוגי החדרים המוצעים על ידי המלון.

**טבלת חיפוש** **ל-CardType:** טבלת חיפוש עבור סוג כרטיס האשראי (Type) מתוך הנחה כי לקוח יכול לשלם רק בכרטיסי אשראי מורשים אשר איתם עובדת החברה.

CREATE TABLE CreditCardType (Types VarChar (30) Primary Key NOT NULL )

INSERT INTO CreditCardType VALUES ('VISA'),('MASTERCARD'),('AMERICAN EXPRESS'),('DIRECT')

ALTER TABLE CreditCards ADD CONSTRAINT FK\_CreditCardType FOREIGN KEY (Types) REFERENCES CreditCardType (Types)

INSERT INTO Hotels(RoomType)

VALUES ('STANDARD'),('GARDEN VIEW'),('SUITE'),('PREMIUM')

## מטלה 1 – שאילתות

**שתי שאילתות SELECT ללא קינון-**

**שאילתה 1:**עבור כל מדינה הצג את הלקוחות שביצעו יותר מ-2 הזמנות ששולמו בכרטיס מסוג "מאסטר קארד" בשנת 2020

יש להציג: מזהה לקוח, מדינה, שם מלא, מספר הזמנה, מסודר לפי מספר הזמנות בסדר יורד.

**תרומה עסקית:** הענקת גיפט קארד לקנייה באתר מחברת מאסטר קארד לרוכשים המזמינים יותר מ-2 חבילות ב2020

SELECT C.Email, C.Country, [Full Name]= C.FirstName + ' ' + C.LastName , [Number Of Orders] = count( \*)

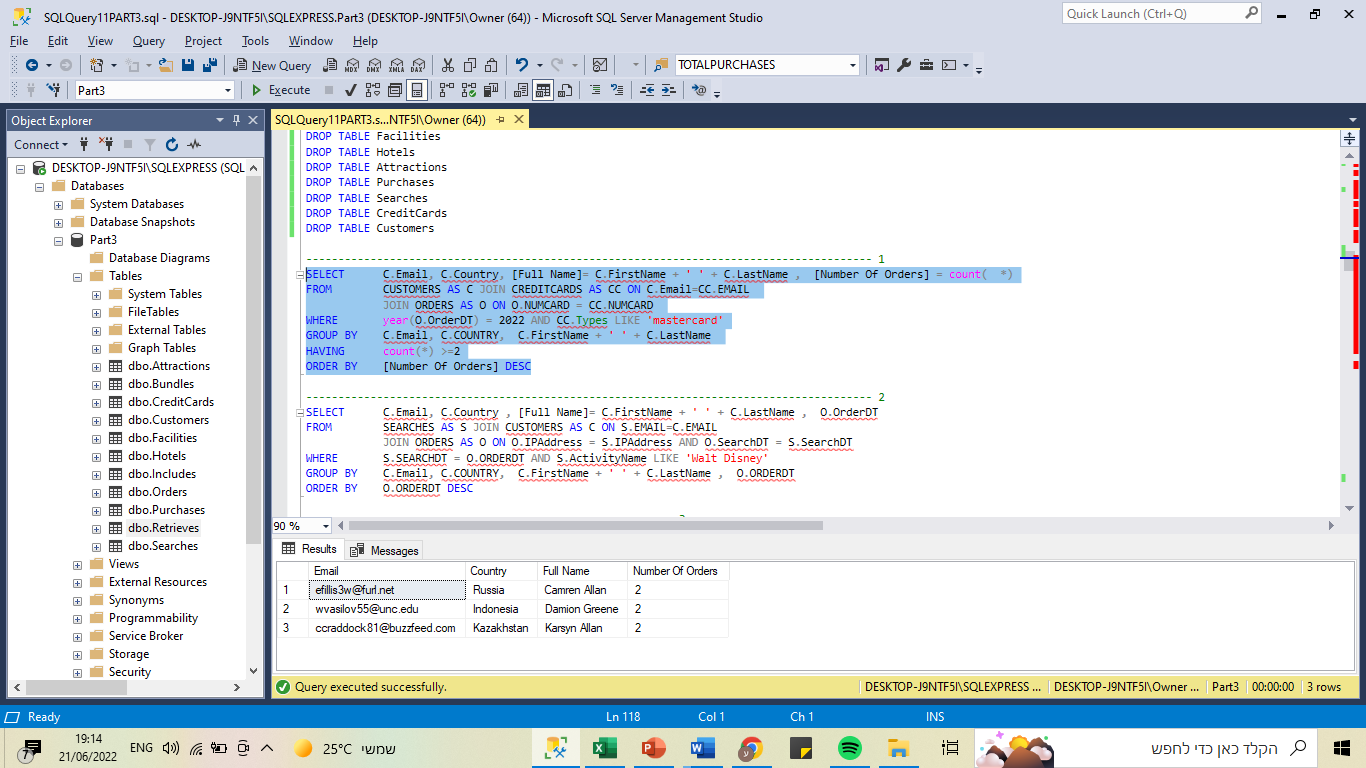
FROM CUSTOMERS AS C JOIN CREDITCARDS AS CC ON C.Email=CC.EMAIL JOIN ORDERS AS O ON O.NUMCARD = CC.NUMCARD

WHERE year(O.OrderDT) = 2022 AND CC.Types LIKE 'mastercard'

GROUP BY C.Email, C.COUNTRY, C.FirstName + ' ' + C.LastName

HAVING count(\*) >=2

ORDER BY [Number Of Orders] DESC

**פלט:**

**שאילתה 2**: יש להציג את כמות ההזמנות "וולט דיסני" מכל המדינות בהן תאריך חיפוש האטרקציה "דיסני וורלד" זהה לתאריך הזמנת האטרקציה "וולט דיסני"

יש להציג: מזהה לקוח, מדינה, שם מלא, מספר הזמנה, מסודר לפי מספר הזמנות בסדר יורד

**תרומה עסקית:** אנשי מכירות רוצים לאמוד את הביקוש לאטרקציות בתאריכים בהם חיפוש מוביל להזמנה באותו היום וכך להעלות או להוריד מחירים בהתאמה – עיקר ההתמקדות (ההכנסות) נעשות מאטרקציית "וולט דיסני".

SELECT C.Email, C.Country , [Full Name]= C.FirstName + ' ' + C.LastName , O.OrderDT

FROM SEARCHES AS S JOIN CUSTOMERS AS C ON S.EMAIL=C.EMAIL

JOIN ORDERS AS O ON O.IPAddress = S.IPAddress AND O.SearchDT = S.SearchDT

WHERE S.SEARCHDT = O.ORDERDT AND S.ATTRACTIONNAME LIKE 'Walt Disney'

GROUP BY C.Email, C.COUNTRY, C.FirstName + ' ' + C.LastName , O.ORDERDT

ORDER BY O.ORDERDT DESC

**תמונה שמכילה טקסט

התיאור נוצר באופן אוטומטי**

**פלט:**

**שתי שאילתות SELECT מקוננות-**

**שאילתה 3 מקוננת -** מחזירה ערך סקלרי.

עבור כל חודש בשנת 2020 בודקת את היחס בין הכנסות חודשיות לסך ההכנסות השנתיות.

יש להציג : חודש , יחס חודשי.

**תרומה עסקית:** מציאת החודש הרווחי ביותר בשנת 2020, בעזרת מידע זה נוכל לקדם הזמנות ולנהל אסטרטגיה עסקית מול לקוחות.

SELECT MonthOrder= MONTH(O.OrderDT), [Ratio Price]= SUM (P.PRICE)/ (

SELECT TOTALPRICE=SUM (P.PRICE)

FROM Includes AS I JOIN orders AS O ON I.OrdersID = O.OrdersID JOIN Purchases AS P ON I.PurchaseID = P.PurchaseID

WHERE YEAR(O.OrderDT)= '2020'

)

FROM Includes AS I JOIN orders AS O ON I.OrdersID= O.OrdersID

JOIN Purchases AS P ON I.PurchaseID= P.PurchaseID

WHERE MONTH(O.OrderDT) BETWEEN 1 AND 12 AND YEAR(O.OrderDT)= '2020'

GROUP BY MONTH(O.OrderDT)

תמונה שמכילה טקסט

התיאור נוצר באופן אוטומטי

**פלט:**

**שאילתה 4 מקוננת –** מחזירה רשימת אטרקציות שחופשו ע"י לקוחות ברזיל שמעולם לא חופשו ע"י לקוחות מפולין.

**תרומה עסקית:** ליצור התאמה עבור כל מדינה ולשווק את האטרקציות האטרקטיביות עבור אותה האוכלוסייה.

SELECT S.ActivityName

FROM Searches AS S JOIN CUSTOMERS AS C ON S.EMAIL=C.EMAIL

WHERE C.COUNTRY='Brazil' AND S.ActivityName NOT IN

(

SELECT DISTINCT S.ActivityName

FROM Searches AS S JOIN CUSTOMERS AS C ON S.EMAIL=C.EMAIL

WHERE C.COUNTRY= 'Poland'

)

GROUP BY S.ActivityName

**תמונה שמכילה טקסט

התיאור נוצר באופן אוטומטיפלט**:

**שתי שאילתות מקוננות תוך שימוש במרכיבים נוספים-**

**שאילתה 5-** שימוש במרכיב UPDATE

עדכן לכל לקוח - האם ביצע הזמנה כלשהי בשנה החולפת.

הוספנו עמודה חדשה בשם Active Customer לטבלת לקוחות שמשמעותה לעדכן האם הלקוח ביצע הזמנה בשנה החולפת. במידה וביצע הזמנה - יתקבל ערך 1, במידה ולא – ערך 0.

**תרומה עסקית –** האתר רוצה לדעת אילו משתמשים היו פעילים בשנה האחרונה ובהתאם לכך לבצע מבצעים והטבות ללקוחות שלא ביצעו הזמנות בשנה החולפת.

--ALTER TABLE CUSTOMERS DROP COLUMN [Active Customer]

ALTER TABLE CUSTOMERS ADD [Active Customer] bit

UPDATE CUSTOMERS SET [Active Customer] =1

WHERE CUSTOMERS.Email IN (

SELECT O.Email

FROM ORDERS AS O

WHERE YEAR(getdate())= YEAR(O.OrderDT)

)

UPDATE CUSTOMERS SET [Active Customer] =0

WHERE CUSTOMERS.Email NOT IN (

SELECT O.Email

FROM ORDERS AS O

WHERE YEAR(getdate())= YEAR(O.OrderDT)

)

SELECT \*

FROM CUSTOMERS AS C

ORDER BY [Active Customer] DESC

**שאילתה 6 –** שימוש באיחוד.

הצג את כל הלקוחות אשר רכשו מלון ואטרקציה בסכום רכישה גדול מסכום הרכישה המקסימלי של השנה שחלפה.

הצג את מייל הלקוח, מחיר הרכישה

**תרומה עסקית –** האתר רוצה להעניק הטבה ללקוחות הרווחים ביותר של השנה הנוכחית.

SELECT O.Email, P.Price

FROM Hotels AS H JOIN Purchases AS P ON H.PurchaseID= P.PurchaseID

JOIN Includes AS I ON P.PurchaseID = I.PurchaseID

JOIN Orders AS O ON O.OrdersID=I.OrdersID

WHERE YEAR(getdate())= YEAR(O.OrderDT) AND P.PRICE > (

SELECT MAX(P.PRICE)

FROM Includes AS I JOIN orders AS O ON I.OrdersID= O.OrdersID JOIN Purchases AS P ON I.PurchaseID= P.PurchaseID

WHERE YEAR(O.OrderDT) = YEAR(getdate())-1

)

UNION

SELECT O.Email, P.Price

FROM Attractions AS A JOIN Purchases AS P ON A.PurchaseID= P.PurchaseID

JOIN Includes AS I ON P.PurchaseID = I.PurchaseID

JOIN Orders AS O ON O.OrdersID=I.OrdersID

WHERE YEAR(getdate())= YEAR(O.OrderDT) AND P.PRICE > (

SELECT MAX(P.PRICE)

FROM Includes AS I JOIN orders AS O ON I.OrdersID= O.OrdersID JOIN Purchases AS P ON I.PurchaseID= P.PurchaseID

WHERE YEAR(O.OrderDT) = YEAR(getdate())-1

)

**פלט:**

תמונה שמכילה שולחן

התיאור נוצר באופן אוטומטי

**מטלה 2 – יישומי כלים מתקדמים**

**View**

**שאילתה 1 –** הצג את עשרת הלקוחות הרווחים ביותר לאתר, שמם המלא, כתובת, וסך המחיר ששילמו.

**תרומה עסקית –** טבלת ה-VIEW תשמש כטבלת עזר לחישוב ההכנסות של הלקוחות לאתר. ניתן לראות שעל מנת להגיע לחישוב שדה זה, התהליך הינו מאוד מורכב ולאחר הקמת טבלת VIEW, יהיה ניתן לשלוף את המידע בצורה נוחה ונגישה יותר.

--DROP VIEW VIEW\_ORDERS

CREATE VIEW VIEW\_ORDERS

AS

SELECT C.Email, [Full Name] = C.FirstName+ ' '+C.LastName, C.Address, [Total Price] = SUM(P.PRICE)

FROM SEARCHES AS S JOIN CUSTOMERS AS C ON S.EMAIL=C.EMAIL

JOIN ORDERS AS O ON O.IPAddress = S.IPAddress AND O.SearchDT = S.SearchDT JOIN

Includes AS I ON I.OrdersID= O.OrdersID

JOIN Purchases AS P ON I.PurchaseID= P.PurchaseID

WHERE C.Email=O.Email

GROUP BY C.Email, C.FirstName+' '+C.LastName, C.Address

**שאילתה:**

SELECT TOP 10 \*

FROM VIEW\_ORDERS

ORDER BY [Total Price] DESC

תמונה שמכילה טקסט

התיאור נוצר באופן אוטומטי

**פלט:**

**פונקציות (Functions)-**

**פונקצייה ראשונה – פונקצייה סקלרית:**

הפונקציה מקבלת שם של אטרקציה וטווח שנים, ומחזירה את מספר הרכישות שבוצעו במהלך השנים הללו.

נעזרנו בטבלת VIEW על מנת לשלוף את כל הנתונים בצורה נוחה יותר.

--DROP VIEW View\_Attractions

CREATE VIEW View\_Attractions

AS

SELECT A.ActivityName, O.OrdersID , O.OrderDT

FROM Attractions AS A JOIN Purchases AS P ON A.PurchaseID= P.PurchaseID

JOIN Includes AS I ON P.PurchaseID = I.PurchaseID

JOIN Orders AS O ON O.OrdersID=I.OrdersID

WHERE O.OrdersID=I.OrdersID and I.PurchaseID=P.PurchaseID

--DROP FUNCTION Num\_Of\_Purchases

CREATE FUNCTION Num\_Of\_Purchases ( @ATTRACTIONNAME VARCHAR (50), @From Datetime, @To Datetime )

RETURNS int

AS BEGIN

DECLARE @Total Int

SELECT @Total = COUNT (V.OrdersID)

FROM View\_Attractions AS V

WHERE v.ActivityName LIKE @AttractionName and YEAR(V.OrderDT) >= @From AND

YEAR(V.OrderDT) <= @To

RETURN @Total

END

**השאילתה:**

SELECT [Amount Of Orders] = dbo.NUM\_OF\_PURCHASES('WonderWorks', 2016,2022)

**תמונה שמכילה טקסט

התיאור נוצר באופן אוטומטיפלט**:

**פונקצייה שנייה – פונקצייה טבלאית:**

הפונקציה מקבלת מזהה לקוח (מייל) המחזירה דו"ח עסקי על רכישתו באתר.  
הדו"ח מציג את מזהה ההזמנה, תאריך ההזמנה, אטרקציות שרכש, דירוג האטרקציה, שם המלון שרכש ואת מחיר של סך הרכישות.

נעזרנו בטבלת VIEW על מנת לשלוף את כל נתוני הלקוח בצורה נוחה יותר.

--DROP VIEW View\_Customer

CREATE VIEW View\_Customer

AS

SELECT C.Email, O.OrdersID,O.OrderDT,A.ActivityName,A.CustomerRate, H.HotelName, P.PRICE

FROM SEARCHES AS S JOIN CUSTOMERS AS C ON S.Email=C.Email

JOIN ORDERS AS O ON O.IPAddress = S.IPAddress AND O.SearchDT = S.SearchDT JOIN

Includes AS I ON I.OrdersID= O.OrdersID

JOIN Purchases AS P ON I.PurchaseID= P.PurchaseID

JOIN Attractions AS A ON A.PurchaseID= P.PurchaseID

JOIN Hotels AS H ON H.PurchaseID= p.PurchaseID

WHERE O.OrderSID=I.OrdersID and I.PurchaseID=P.PurchaseID

-- DROP FUNCTION Customer\_Report

CREATE FUNCTION Customer\_Report (@CustomerEmail Varchar(50))

RETURNS TABLE

AS RETURN

SELECT V.OrdersID,V.OrderDT,V.ActivityName,V.CustomerRate, V.HotelName, V.Price

FROM View\_Customer AS V

WHERE V.Email = @CUSTOMEREmail

GROUP BY V.OrdersID,V.OrderDT,V.ActivityName,V.CustomerRate, V.HotelName, V.Price

**שאילתה:**

SELECT \*

FROM dbo.Customer\_Report('efillis3w@furl.net')

ORDER BY OrderDT

**תמונה שמכילה טקסט

התיאור נוצר באופן אוטומטיפלט:**

**Trigger פשוט –**

עדכון כמות רכישות שבוצעו על ידי כל לקוח

הוספנו עמודת סה"כ רכישות לטבלת ORDERS, הטריגר מעדכן אוטומטית את מספר הרכישות הכולל של כל הלקוחות כאשר נוספת\משתנה\נמחקת הזמנה כלשהי.

הטריגר מאפשר לנו בכל נקודת זמן לדעת כמה רכישות ביצע כל לקוח.

--ALTER TABLE CUSTOMERS DROP column TotalPurchases;

ALTER TABLE CUSTOMERS ADD TotalPurchases int

--DROP TRIGGER Update\_Orders

CREATE TRIGGER Update\_Orders

ON ORDERS

FOR INSERT, UPDATE, DELETE

AS BEGIN

UPDATE Customers SET totalPurchases = (

SELECT COUNT(\*)

FROM Orders

WHERE Customers.Email=Orders.Email

)

WHERE Email IN (

SELECT DISTINCT Email FROM INSERTED

UNION

SELECT DISTINCT Email FROM DELETED

)

END

INSERT INTO Searches VALUES ('3.195.208.163', '2020-08-20 00:00:00.000', 'Meal Tour', 'efillis3w@furl.net')

INSERT INTO Orders VALUES ('60821', 'efillis3w@furl.net', '1795966839332760', '2020-08-20 00:00:00.000', '3.195.208.163', '2020-08-20 00:00:00.000')

**שאילתה:**

SELECT \*

FROM Customers AS C

WHERE C.Email = 'efillis3w@furl.net'

תמונה שמכילה טקסט

התיאור נוצר באופן אוטומטי**פלט לפני:**

**תמונה שמכילה טקסט

התיאור נוצר באופן אוטומטיפלט אחרי:**

**פרוצדורה שמורה ((Stored Procedure פשוטה –**

פרוצדורה הנותנת ללקוח לדרג בשנית את האטרקציה שדירג. הפרוצדורה מקבלת 3 משתנים - מייל הלקוח, הדירוג החדש אותו ירצה לדרג ומספר ההזמנה של הלקוח.

צורך עסקי – במידה והלקוח טעה בדירוג הראשוני שלו ורוצה לתקן, תהיה האפשרות לתקן זאת.

--DROP VIEW View\_CustomerRate

CREATE VIEW View\_CustomerRate

AS

SELECT C.Email ,A.CustomerRate

FROM SEARCHES AS S JOIN CUSTOMERS AS C ON S.EMAIL=C.EMAIL

JOIN Orders AS O ON O.IPAddress = S.IPAddress AND O.SearchDT = S.SearchDT

JOIN Includes AS I ON I.OrdersID= O.OrdersID

JOIN Purchases AS P ON I.PurchaseID= P.PurchaseID

JOIN Attractions AS A ON A.PurchaseID= P.PurchaseID

WHERE O.OrderSID=I.OrdersID and I.PurchaseID=P.PurchaseID

--DROP PROCEDURE SP\_Rank\_CustomerRate

CREATE PROCEDURE SP\_Rank\_CustomerRate

@CustomerEmail Varchar(50), @newRate tinyInt , @PurchaseID int

AS BEGIN

UPDATE Attractions SET CustomerRate = @newRate

WHERE PurchaseID = @PurchaseID

END

RETURN

**הפעלת הפרוצדורה:**

EXECUTE SP\_RANK\_CustomerRate 'abannister3y@hhs.gov' ,'1', '10401'

**שאילתה:**

SELECT \*

FROM View\_CustomerRate AS V

**פלט לפני הפעלה:**

תמונה שמכילה טקסט

התיאור נוצר באופן אוטומטי

**פלט אחרי הפעלה:**

תמונה שמכילה טקסט

התיאור נוצר באופן אוטומטי

## מטלה 3 – כלים להצגת נתונים

**דוח עסקי:**

הדו"ח הבא מיועד למחלקת שיווק של האתר "Attraction Tickets" .   
המטרה העסקית של הדו"ח היא הצגת נתונים כלכליים כגון: כמות הזמנות ,דירוגי האטרקציות והרווח הכלכלי של האתר מהן.  
הנתונים יציגו מידע לגבי מספר ההזמנות לפי מדינת הפעילות , סך ההכנסות של השנה החולפת לפי חודשים, פילוח של החבילות על פי מספר הרכישות והצגת האטרקציות על פי דירוגם.  
על סמך בסיס הנתונים הנ"ל, מחלקת השיווק תוכל לקבל החלטות כלכליות ושיווקיות לדוגמה באילו מדינות (בהן קיימות פעילויות) הביקוש לפעילויות גבוה מהאחרים, אילו אטרקציות מדורגות גבוה ביחס לאחרות וכדומה.

**שאלה 1:** באילו מדינות בשנה האחרונה מספר הרכישות הוא הגבוה ביותר?

**תרומה עסקית:** סיפוק מידע אילו מדינות יותר אטרקטיביות וכך להגביר את ההיצע של הפעילויות המוצעות באתר במדינות אלו.

בעת יצירת הדו"ח בנינו את ה VIEW הבא:

--DROP VIEW VIEW\_Num\_Of\_Orders

CREATE VIEW VIEW\_Num\_Of\_Orders

AS

SELECT P.Country, [Number Of Orders]=COUNT(\*)

FROM Purchases AS P

JOIN Includes AS I ON P.PurchaseID = I.PurchaseID

JOIN Orders AS O ON O.OrdersID=I.OrdersID

WHERE YEAR(O.OrderDT)=YEAR(GETDATE())  
 GROUP BY P.Country

**שאלה 2:** מהו ממוצע דירוג כל אטרקציה בשנה האחרונה ?

**תרומה עסקית:** מציג למחלקת השיווק אילו אטרקציות מובילות בדירוגן ואילו פחות ועל ידי כך למנף את השיווק של כל אחת מהן, ומסייע בקבלת אסטרטגיות תפעוליות באתר.

בעת יצירת הדו"ח בנינו את ה VIEW הבא:

--DROP VIEW VIEW\_AttractionRate

CREATE VIEW VIEW\_AttractionRate

AS

SELECT A.ActivityName, [Average Rate] = AVG(A.CustomerRate)

FROM Includes AS I JOIN Orders AS O ON O.OrdersID=I.OrdersID JOIN Purchases AS P ON P.PurchaseID = I.PurchaseID JOIN Attractions AS A ON A.PurchaseID=P.PurchaseID

WHERE YEAR(O.OrderDT)=YEAR(GETDATE())  
GROUP BY A.ActivityName

**שאלה 3:** באילו חודשים בשנה האחרונה סכום הרכישות הוא הגבוה ביותר?

**תרומה עסקית:** מספק מידע אילו חודשים הם הרווחיים ביותר עבור האתר ובכך למנף את הרווח הכלכלי של האתר בחודשים אלו.

בעת יצירת הדו"ח בנינו את ה VIEW הבא:

--DROP VIEW VIEW\_Purchase\_Price

CREATE VIEW VIEW\_Purchase\_Price

AS

SELECT Month=month(O.OrderDT), [Revenue by Month] = SUM(p.price)

FROM Includes AS I JOIN Orders AS O ON O.OrdersID=I.OrdersID JOIN Purchases AS P ON P.PurchaseID = I.PurchaseID JOIN Attractions AS A ON A.PurchaseID=P.PurchaseID

WHERE YEAR(O.OrderDT)=YEAR(GETDATE())  
GROUP BY MONTH(O.OrderDT),YEAR(O.OrderDT)

**שאלה 4:** מציג את סך הרכישות שהתבצעו עבור כל חבילה.

**תרומה עסקית:** מספק למחלקת השיווק מידע עבור אילו חבילות נרכשות יותר ואילו פחות ועל ידי כך להתאים את אסטרטגיית השיווק בהתאם לכל חבילה.

--DROP VIEW Number\_Of\_Bundle\_Orders

CREATE VIEW Number\_Of\_Bundle\_Orders

AS

SELECT Bundle = B.BundleName, [Number of Orders] = COUNT (\*)

FROM Purchases AS P

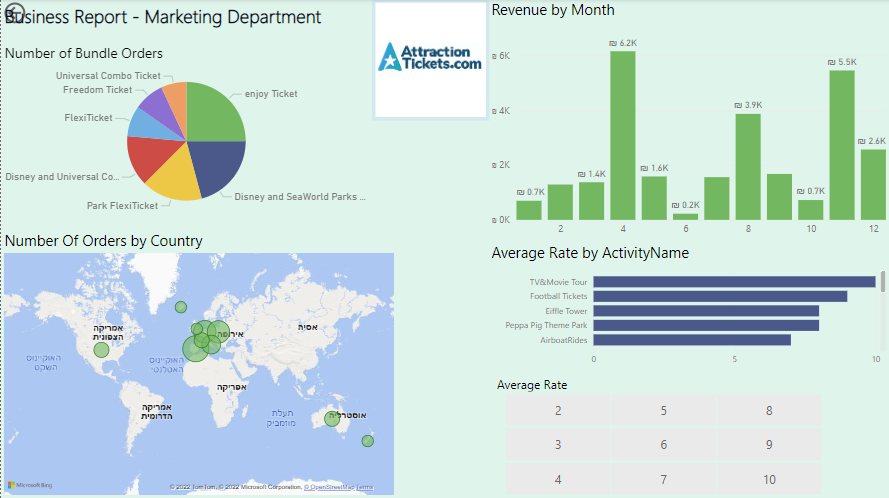
JOIN Attractions AS A ON A.PurchaseID= P.PurchaseID

JOIN Bundles as B on B.BundleName= a.ActivityName

JOIN Includes as I on I.PurchaseID = P.PurchaseID

JOIN Orders as O on O.OrdersId = I.OrdersID

GROUP BY B.BundleName



**לוח מחוונים**

דוח זה מיועד למנהלים הבכירים.

מטרת הדוח- הצגת נתונים ברמה גרעינית נמוכה על מנת לסייע למנהלים לקבל החלטות מכריעות בנוגע לעתיד האתר.

לשם יצירת לוח המחוונים בנינו VIEW שעליו מתבססים רוב התרשימים המוצגים בלוח:

--DROP VIEW VIEW\_REPORT

CREATE VIEW VIEW\_REPORT

AS

SELECT CC.Email, CC.FirstName, CC.LastName, CC.Country, S.IPAddress, S.SearchDT, [Search by Activity Name] = S.ActivityName, C.Types, O.OrdersID, O.OrderDT, P.PurchaseID, [Country of Activity] = P.Country, P.Price, [Purchase by Activity Name] = A.ActivityName, A.CustomerRate, H.HotelName, H.RoomType

FROM ORDERS AS O JOIN CreditCards AS C ON O.NUMCARD = C.NUMCARD join Customers as CC on CC.Email=o.Email

JOIN SEARCHES AS S ON O.IPAddress = S.IPAddress AND O.SearchDT = S.SearchDT

JOIN Retrieves AS R ON R.IPAddress = S.IPAddress AND R.SearchDT = S.SearchDT

JOIN Purchases AS P ON R.PurchaseID= P.PurchaseID

JOIN Attractions AS A ON A.PurchaseID= P.PurchaseID JOIN Hotels AS H ON H.PurchaseID= p.PurchaseID

**נתוני הדוח:**

* **נתונים כמותיים:** המציגים את מספר ההזמנות שבוצעו באתר, ואת סך ההכנסות.
* **מספר הלקוחות באתר על פי ארצות-**  מציג את כמות הלקוחות בכל מדינה ובכך יסייע למנהלים להבין את קהל היעד העיקרי של האתר.
* **מחיר הזמנה לפי שנה- תרשים גרף:** מציג את סכום הרכישות לכל שנה.
* **רווח כל אטרקציה עבור האתר- כתרשים עמודות:** מציג את סכום ההכנסות של כל אטרקציה לאתר. יספק למנהלים מידע באילו אטרקציות להשקיע יותר, אילו אטרקציות כדאי להחליף וכו'.
* **מספר הזמנות/הכנסות והמטרות בהתאם**-   
  היעד הוא הגדלת מספר ההזמנות ב15 אחוזים.  
  בנוסף, הגדלת ההכנסות מהשנה שחלפה ב5 אחוזים.

--DROP VIEW V\_ TargetForOrder

CREATE VIEW V\_TargetForOrder

AS

SELECT [Target]= ROUND (1.15 \* COUNT(o.OrdersID),0)

FROM Orders as O

WHERE YEAR(o.OrderDT)=YEAR(GETDATE()-1)

--DROP VIEW V\_sumOfOrder

CREATE VIEW V\_sumOfOrder

AS

SELECT COUNT(o.OrdersID) AS sum1

FROM Orders as O

WHERE YEAR(o.OrderDT)=YEAR(GETDATE()-1)

--DROP VIEW V\_TargetForRevenue

CREATE VIEW V\_TargetForRevenue

AS

SELECT [Target] = 0.85 \* SUM(P.Price)

FROM Includes AS I JOIN Orders AS O ON O.OrdersID=I.OrdersID JOIN Purchases AS P ON P.PurchaseID = I.PurchaseID

WHERE YEAR (O.OrderDT)= YEAR(GETDATE()-1)

--DROP VIEW V\_sumRevenue

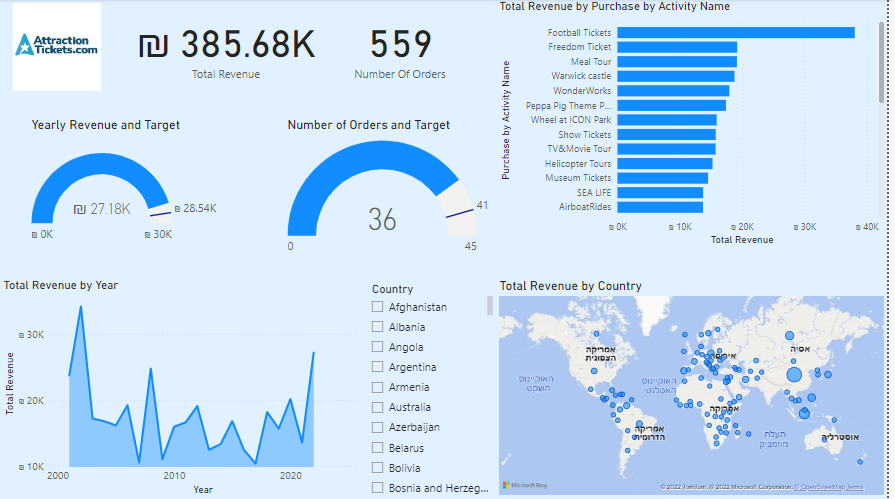
CREATE VIEW V\_sumRevenue

AS

SELECT [Current Year] = SUM(P.Price)

FROM Includes AS I JOIN Orders AS O ON O.OrdersID=I.OrdersID JOIN Purchases AS P

ON P.PurchaseID = I.PurchaseID  
WHERE YEAR(o.OrderDT)=YEAR(GETDATE())



## מטלה 4 – יישומים מתקדמים

**שאילתות עסקיות המשלבות Window Functions**

**שאילתה 1:** השתמשנו בפונקציית Lag(), השאילתה מציגה את יחס ההכנסות בין השנה הנוכחית לבין השנה הקודמת.

**תרומה עסקית:** הפקת נתונים על הכנסות האתר ביחס לשנה שחלפה.

SELECT V.Year, V.[Sum Per Year], [Growth By Percentage] = (V.[Sum Per Year] / LAG (V.[Sum Per Year]) OVER (ORDER BY YEAR)) -1

FROM (

SELECT [Year] = YEAR(O.OrderDT), [Sum Per Year] = SUM (P.Price)

FROM Includes AS I JOIN Orders AS O ON O.OrdersID =I.OrdersID JOIN Purchases AS P ON P.PurchaseID = I.PurchaseID

GROUP BY YEAR(O.OrderDT)

) AS V  
תמונה שמכילה שולחן

התיאור נוצר באופן אוטומטי

**פלט:**

**שאילתה 2:** השתמשנו בפונקציות DENSE RANK(),Percent Rank() ו- Cume\_Dist(), השאילתה מציגה את החלק היחסי של סך ההכנסות של כל לקוח מסך ההכנסות של כל הלקוחות מאותה המדינה.   
עבור Percent Rank () , הפונקציה מציגה עבור כל לקוח אחוזון לפי מחיר.  
עבור Cume Dist (), לפי פונקציית התפלגות מצטברת, כלומר את המיקום היחסי של כל לקוח מכלל הלקוחות מאותה המדינה ביחס למחיר.  
עבור Dense Rank(), מטרתה לדרג כל לקוח לפי שיעור ההכנסה היחסי מכלל הלקוחות באותה המדינה.

**תרומה עסקית:** ניתוח מעמיק של ניהול תיק הלקוחות ושימור הלקוחות הרווחים ביותר מכל מדינה.

SELECT DISTINCT X.[Full Name], X.Country, V.Email, V.[Sum Per Customer],

[Percent By Country] = PERCENT\_RANK () OVER (PARTITION BY X.Country ORDER BY V.[Sum Per Customer]),

[Cume Dist] = CUME\_DIST() OVER (PARTITION BY X.Country ORDER BY V.[Sum Per Customer]) ,

[Rank] = DENSE\_RANK() OVER (PARTITION BY X.Country ORDER BY V.[Sum Per Customer] )

FROM (

SELECT [Full Name] = C.FirstName + ' ' + C.LastName, C.Country , C.Email

FROM SEARCHES AS S JOIN CUSTOMERS AS C ON S.EMAIL=C.EMAIL

JOIN ORDERS AS O ON O.IPAddress = S.IPAddress AND O.SearchDT = S.SearchDT JOIN Includes AS I ON I.OrdersID= O.OrdersID

JOIN Purchases AS P ON I.PurchaseID= P.PurchaseID

) AS X JOIN

(

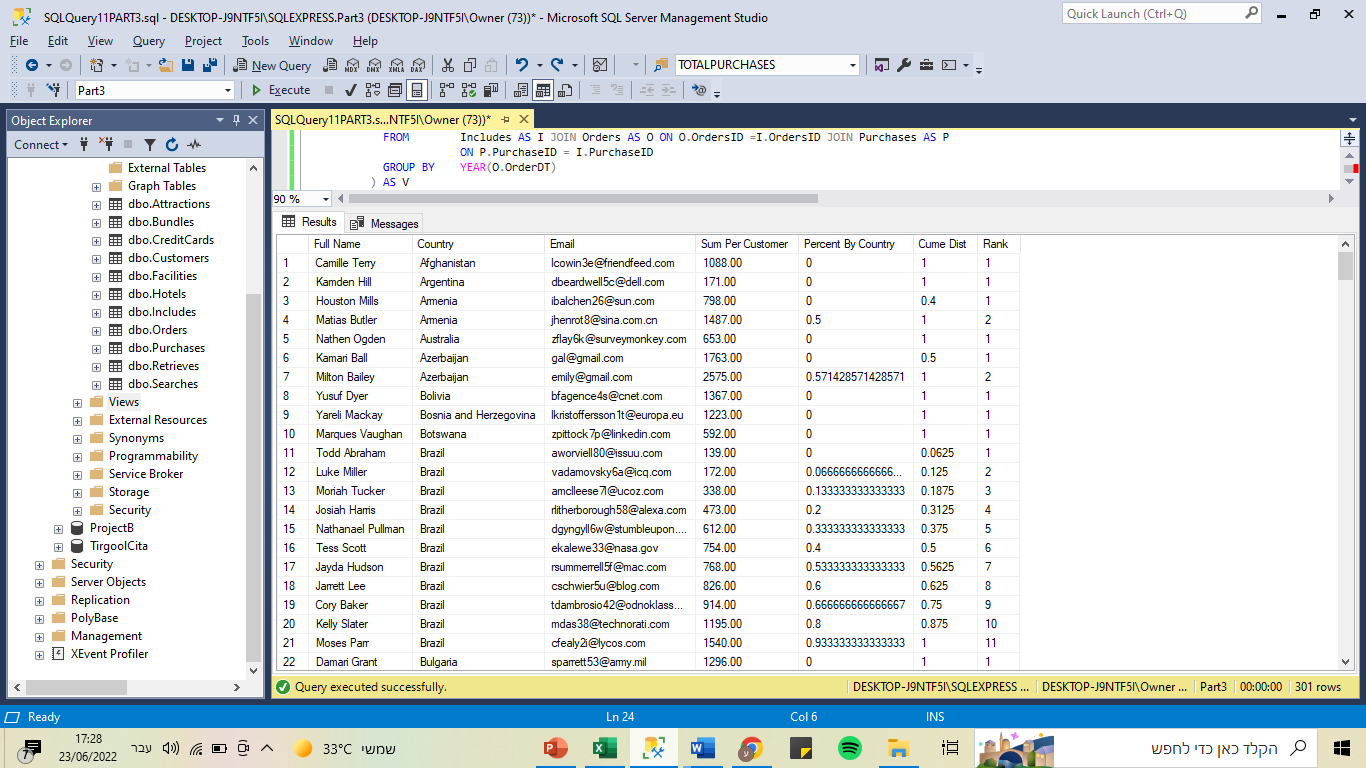
SELECT C.Email, [Sum Per Customer] = SUM(P.Price)

FROM SEARCHES AS S JOIN CUSTOMERS AS C ON S.EMAIL=C.EMAIL JOIN ORDERS AS O ON O.IPAddress = S.IPAddress AND O.SearchDT = S.SearchDT JOIN Includes AS I ON I.OrdersID= O.OrdersID JOIN Purchases AS P ON I.PurchaseID= P.PurchaseID

GROUP BY C.Email

) AS V ON X.Email = V.Email

ORDER BY X.Country, V.[Sum Per Customer] DESC



**פלט:**

**שילוב מערכתי של מספר כלים –**

פרוצדורה שמקבלת טווח תאריכים ויוצרת טבלה שמראה את כל הדירוגים שהיו לאטרקציות באותם תאריכים.   
עבור כל אטרקציה הטבלה מראה את ממוצע הדירוג הנוכחי ואת הדירוג האחרון בטווח התאריכים שהוזנו.

חישוב הדירוג הקודם עבור האטרקציות מתבצע על ידי פונקציה.  
בנוסף, בטבלה הנוצרת יצרנו עמודה נוספת שהיא מחשבת את אחוז השיפור (החיובי או השלילי) בין ממוצע הדירוג הנוכחי לדירוג הקודם.

יצרנו VIEW על מנת לחשב את ממוצע הדירוגים עבור כל אטרקציה.  
לבסוף הפרוצדורה משתמשת בview על מנת להציג רק את האטרקציות אשר היה שינוי בדירוג שקיבלו.  
  
**תרומה עסקית:** הענקת מידע עבור הנהלת האתר לגבי אחוז השינוי (החיובי או השלילי) של האטרקציות בטווח תקופות מסוימות, על מנת ליצור התאמה מדויקת לגבי אילו אטרקציות מתאימות יותר לשביעות רצונם של הלקוחות בתקופות שונות ולקדמן.

--------------procedure

--drop procedure sp\_Rate

create procedure sp\_Rate(@fromDate datetime, @toDate datetime)

as

if (select OBJECT\_ID ('attractionRate')) is not null drop table attractionRate

create table attractionRate (

activityName varchar(50) not null,

currentRate tinyInt null,

previousRate tinyInt null,

Improvement real null,

primary key (activityName )

)

insert into attractionRate select activityName=v.activityname,

currentRate =v.avgcustomerRate,

previousRate = (select top 1 rateByDate.previousRate

from dbo.previousRate(v.activityName, @fromDate,@toDate) as rateByDate

order by rateByDate.orderDate DESC ),

Improvement = case when

(select top 1 rateByDate.previousRate

from dbo.previousRate(v.activityName, @fromDate ,@todate) as rateByDate

order by rateByDate.orderDate DESC ) != 0 then

cast(cast(v.avgcustomerRate as int) -

(select top 1 rateByDate.previousRate

from dbo.previousRate(v.activityName, @fromDate,@toDate) as rateByDate

order by rateByDate.orderDate DESC ) as real) /

(select top 1 rateByDate.previousRate

from dbo.previousRate(v.activityName, @fromDate,@todate) as rateByDate

order by rateByDate.orderDate DESC )

else 0 end

from v\_Attractions as v

select \*

from v\_ans

-----function

drop function previousRate

create function previousRate (@activityName varchar(50), @fromDate datetime, @toDate datetime)

returns table

as return

select LastorderDT = o.orderDT, previousRate = a.customerRate

FROM Includes AS I JOIN Orders AS O ON O.OrdersID=I.OrdersID JOIN Purchases AS P

ON P.PurchaseID = I.PurchaseID JOIN Attractions AS A ON A.PurchaseID=P.PurchaseID

where A.activityName = @activityName and (o.orderDT between @fromDate and

dateadd(day, -1, @toDate)

)

-------------view

drop view v\_ans

create view v\_ans as

select \*

from attractionRate as s

where s.Improvement != 0

-------------------------------

drop view v\_Attractions

create view v\_Attractions as

select a.Activityname,avgcustomerRate=avg(a.customerRate)

from Includes AS I JOIN Orders AS O ON O.OrdersID=I.OrdersID JOIN Purchases AS P

ON P.PurchaseID = I.PurchaseID JOIN Attractions AS A ON A.PurchaseID=P.PurchaseID

GROUP by a.Activityname

---מימוש

execute sp\_Rate '2004-06-01 00:00:00.000', '2006-08-01 00:00:00.000'

תמונה שמכילה שולחן

התיאור נוצר באופן אוטומטי

## דו"ח המושתת על שימוש בפסקת WITH מורכבת-

הכנסות האתר לפי מדינה לשנת 2022 – סכימה של הכנסות האתר מחולק לפי מדינות של הלקוחות הקיימים באתר.  
האם יש מגמת שיפור ביחס לשנה הקודמת (1), אחרת (0) – בחינה האם הייתה עלייה או ירידה בהכנסות האתר ביחס לשנה שחלפה.  
כמות הזמנות לפי מדינה לשנת 2022 – ספירת כמות ההזמנות שנכנסו לאתר בשנת 2022.  
תרומה ממוצעת של לקוח להכנסות האתר מכל מדינה- בחינת החלק היחסי של כל לקוח להכנסות האתר לפי מדינתו.

**תרומה עסקית:** לספק למנהלי האתר מידע אודות הכנסות האתר לפי מדינות הלקוחות והאם הייתה מגמת עלייה או ירידה ביחס לשנה הקודמת.  
בכך המנהלים יוכלו לנקוט באמצעים אסטרטגיים לגבי המדינות הממוקמות בתחתית המדינות המכניסות וכן, לשמר ולהעצים את מבחר היצע המדינות המכניסות ביותר.

with revenueByCountry as ( select c.Country, REVENUE=sum(p.price)

from SEARCHES AS S JOIN CUSTOMERS AS C ON S.EMAIL=C.EMAIL

JOIN Orders AS O ON O.IPAddress = S.IPAddress AND O.SearchDT = S.SearchDT

JOIN Includes AS I ON I.OrdersID= O.OrdersID

JOIN Purchases AS P ON I.PurchaseID= P.PurchaseID

where year (o.orderDT)= '2022' and O.ordersID= I.ORDERSID AND P.PURCHASEID= I.PURCHASEID

group by C.Country ),

Improve as ( select x.Country ,Improve = CASE WHEN x.revenue< y.revenue THEN 1 ELSE 0 END

from (select c.Country, REVENUE=sum(p.price)

from SEARCHES AS S JOIN CUSTOMERS AS C ON S.EMAIL=C.EMAIL

JOIN Orders AS O ON O.IPAddress = S.IPAddress AND O.SearchDT = S.SearchDT

JOIN Includes AS I ON I.OrdersID= O.OrdersID

JOIN Purchases AS P ON I.PurchaseID= P.PurchaseID

where year (o.orderDT)= '2021' and O.ordersID= I.ORDERSID AND P.PURCHASEID= I.PURCHASEID

group by c.country ) as x join

( select \*

from revenueByCountry

) as y on x.Country= y.Country ),

NumberOfOrders AS (SELECT C.Country , NumberOfOrders = count(\*)

From SEARCHES AS S JOIN CUSTOMERS AS C ON S.EMAIL=C.EMAIL

JOIN ORDERS AS O ON O.IPAddress = S.IPAddress AND O.SearchDT = S.SearchDT

WHERE year(O.OrderDT) = '2022'

GROUP BY C.Country

),

AvgPriceByCustomer AS ( select M.country, AvgPriceByCustomer= (N.REVENUE / [Num Of Customer])

From (SELECT C.Country, [Num Of Customer]= count ( distinct O.email)

From SEARCHES AS S JOIN CUSTOMERS AS C ON S.EMAIL=C.EMAIL

JOIN Orders AS O ON O.IPAddress = S.IPAddress AND O.SearchDT = S.SearchDT

JOIN Includes AS I ON I.OrdersID= O.OrdersID

where year(O.OrderDT) = '2022'

GROUP BY C.Country ) as M join

( select \*

from revenueByCountry

) as N on M.Country= N.Country )

-----הדוח-------------------------

SELECT Country = RBC.country,

Revenue= RBC.Revenue,

Improve=I.Improve ,

[Num Of Orders]= N.NumberOfOrders,

[Avg Price By Customer]=A. AvgPriceByCustomer

FROM revenueByCountry AS RBC JOIN NumberOfOrders as N on RBC.Country= N.Country

JOIN Improve AS I on I.country=RBC.COUNTRY

JOIN AvgPriceByCustomer AS A on A.country = RBC.country

תמונה שמכילה שולחן

התיאור נוצר באופן אוטומטי

**פלט:**