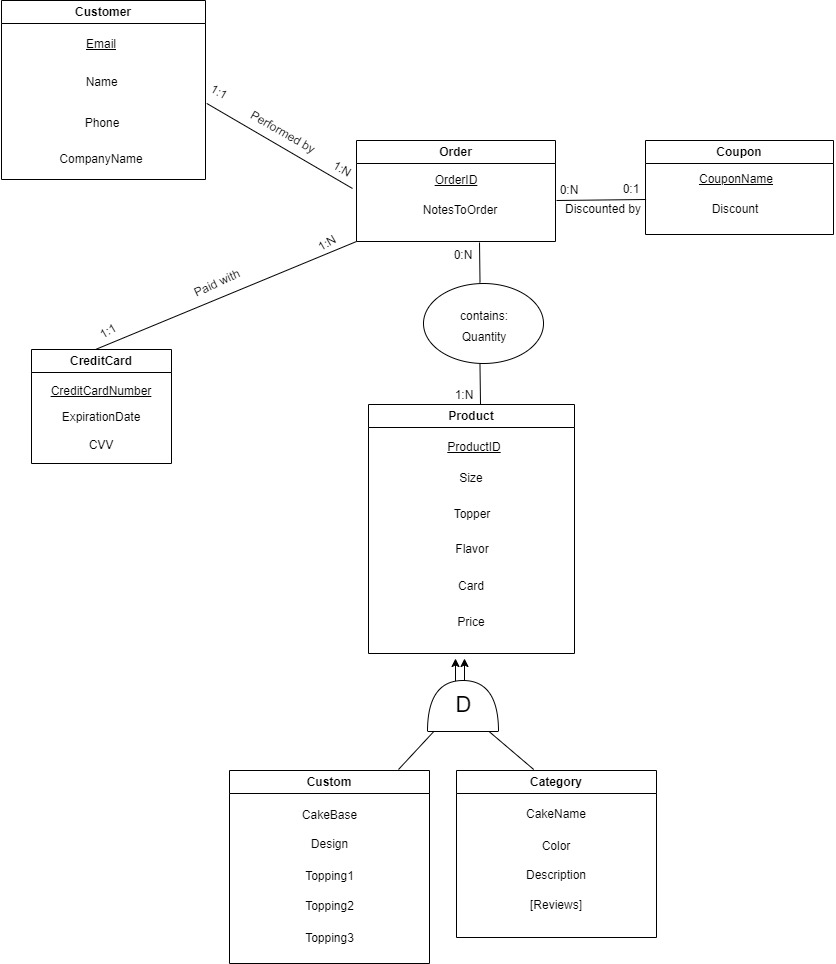
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| מס' קבוצה | אתר | | | תאריך הגשה |
| 28 | https://ruwiscakes.com.au/ | | | 10/03/2024 |
| חברי הצוות - מספרי ת.ז | | | | |
| רותם דורון 208579813 | | עדן חברי 211701396 | יואב סלע 208935601 | |

## פרויקט בסיסי נתונים – חלק ג'

## פרק ראשון

# מטלת חובה מקדימה – תיקון ה-ERD והרחבת בסיס הנתונים

**התרשים בהגשה הקודמת**



**המודל הטבלאי בהגשה הקודמת**

Customers(Email, FirstName, LastName, Phone, CompanyName)

CreditCards(CreditCardNumber, ExpirationDate, CVV)

Products(ProductID, Size, Topper, Flavor, Card, Price)

Customs(ProductID (Products), CakeBase, Design, Topping1, Topping2, Topping3)

Categories(ProductID (Products), CakeName, Color)

Reviews(ProductID(Category), Review)

Coupons(CouponName, Discount)

Orders(OrderID, NotesToOrders, Customer (Customers), CreditCard(CreditCards), CouponName(Coupons))

Contains(ProductID(Products), OrderID(Orders), Quantity)

**הערות שנתנו בהגשה הקודמת**

* בישות Customer: שם הינו תכונה מורכבת, נדרש סימון עבור תכונה מורכבת לדוגמא: first-name.
* בקשר בין Customer ל Order: לפי הקרדינליות, לקוח רשום ביצע לפחות הזמנה 1. נדרשת לכך הנחה או תיקון לn:0 .
* בישות Order: חסרה הכתובת למשלוח.
* בישות :Coupon הנחה היא שדה מחושב ולכן לא נכללת במודל, ניתן ליצור קופון בתוך ההזמנה.
* בישות Product: SIZE, TOPPERS אלו תכונות הניתנות לעיצוב על ידי הלקוח חישבו כיצד לייצג ישות בעלת תכונות שאינן ניתנות לעיצוב וישות בעלת תכונות אשר ניתנות לעיצוב.
* בישות Product: PRICE אינו יכול להיות מחיר משוקלל, מאחר ושדות מחושבים אינם מיוצגים במודל ה ERD אם המחיר נקבע על פי סוג העוגה הוא צריך להופיע כתכונה בישות עם מזהה סוג העוגה והתוספת למחיר צריכה להופיע כשדה, מתוך כך בהמשך יהיה ניתן לחשב שדה מחושב.
* בישות בן Custom: גם עבור עוגות שניתנות לעיצוב יש תכונה cakename.
* בישות בן Category: אם יש reviews למוצר באתר יש לייצגו כישות נפרדת ולא כתכונה מרובת ערכים. כך שיהיה ניתן לזהות מי הלקוח שכתב את חוות הדעת.
* היה צורך להוסיף את קטעי הקוד לכתיבת האילוצים לצד ההסבר.
* ניתן לבצע כמות מינימאלית להזמנה, מאחר והאתר עוסק בהזמנה סיטונאית, למשל מעל 100.

## עיצוב קונספטואלי (תרשים ישויות-קשרים חדש)

## 

**הנחות מודל ה ERD:**

* הכתובת מקושרת להזמנה ולא ללקוח משום שלקוח יכול להזמין משלוחים לכתובות שונות ולא בהכרח לכתובת המגורים שלו.
* בישות - Coupon  
  1. השדה DiscountPrecent מייצג את שיעור ההנחה שהקופון מקנה באחוזים.
* 2. בכל הזמנה ניתן לממש עד קופון אחד.
* הכללנו את שיטת התשלום לכרטיס אשראי בלבד כפי שנאמר בהנחיות.
* בישות אב - Product

1. המזהה DesignID מזהה ייחודי שמתחשב בבחירות הלקוח לפי שאר השדות.
2. BasePrice הוא מחיר הבסיס לעוגה (שנקבע ע"פ סוג העוגה ללא התוספות).
3. AdditionalPrice הוא התוספת למחיר הבסיס שנקבעת לפי בחירות הלקוח (לא המחיר המשוכלל אלא רק התוספת שיש להוסיף למחיר הבסיס).
4. בחרנו שההורשה תהיה מסוג Total-Disjoint משום שעוגה יכולה או להיבנות מאפס או להיבחר מהקטלוג.

* ישות בן Category מתייחסת לעוגות שנבחרות מתוך הקטלוג.
* ישות בן Custom -

1. הישות מתייחסת לעוגות שנבנות מאפס ע"י הלקוח (לעוגות אלה אין שם).
2. השדה CakeIcing מייצג את השלב של בחירת ציפוי העוגה.
3. ניתן לבחור את אותו Topping מספר פעמים לאותה עוגה.

* הקשר Write מייצג את הביקורות שלקוח רושם על עוגה מהקטלוג, קשר הוא מסוג רבים לרבים משום שלקוח אחד יכול לרשום ביקורות על כמה עוגות או לא לרשום בכלל (ולכן:N 0(  
  ועל כל עוגה כמה לקוחות יכולים לכתוב ביקורת לא שאף לקוח לא ירשום ביקורת (ולכן:N 0(.

**עיקר התיקונים שנעשו-**

* סימון של שם הלקוח כשדה מורכב.
* הוספת כתובת כשדה מורכב ליישות הזמנה.
* פירוק המחיר למחיר בסיס ותוספת למחיר.
* שינוי של שדה מרובה ערכים Reviews לקשר.

## עיצוב לוגי (מודל נתונים טבלאי)

Customers(Email, Name-First, Name-Last, Phone, CompanyName)

CreditCards(CreditCardNumber, ExpirationDate, CVV)

Products(DesignID, Size, Topper, Flavor, Card, BasePrice, AdditionalPrice)

Customs(DesignID(Products), CakeBase, CakeIcing, Topping1, Topping2, Topping3)

Categories(DesignID(Products), CakeName, Color)

Coupons(CouponName, DiscountPercent)

Orders(OrderID, NotesToOrders, Address-Country, Address-City, Address-Street, Address-HouseNumber, Address-Apartment, Customer(Customers), CreditCard(CreditCards), CouponName(Coupons))

Contains(DesignID(Products), OrderID(Orders), Quantity)

Writes(Customer(Customers), Cake(Categories), Review)

## מטלה 1 – שאילתות

# שתי שאילתות SELECT ללא קינון

**שאילתה מספר 1:**

השאילתה מציגה לכל שיעור הנחה את סכום ההכנסה הכולל מההזמנות (לאחר ניכוי ההנחה) שמומש בהן קופון שמקנה את אחוז ההנחה הנ"ל.  
השאלה העסקית: כמה כסף מכניס לחברה כל שיעור הנחה?

שאלה זו עשויה להיות רלוונטית כדי להחליט מהם שיעורי ההנחה הכי משתלמים לחברה, ולפיכך להחליט באיזה שיעור הנחה להשתמש יותר/ פחות.

SELECT DiscountPercent,

TotalRevenue = ROUND(SUM(Quantity \* (BasePrice + ISNULL(P.AdditionalPrice, 0)) \* (1-DiscountPercent)),0)

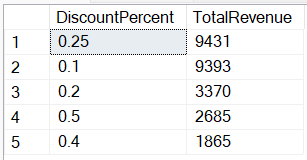
FROM Coupons JOIN Orders AS O ON Coupons.CouponName = O.CouponName

JOIN [Contains] ON O.OrderID = [Contains].OrderID

JOIN Products AS P ON P.DesignID = [Contains].DesignID

GROUP BY DiscountPercent

ORDER BY ROUND(SUM(Quantity \* (BasePrice + ISNULL(P.AdditionalPrice, 0)) \* (1-DiscountPercent)),0) DESC



**שאילתה מספר 2:**

השאלה העסקית: איזה לקוחות ביצעו יותר מ3 הזמנות?

עבור לקוחות שעומדים בתנאי (מספר ההזמנות גדול מ3) השאילתה מציגה את מזהה הלקוח, שם מלא וכמות ההזמנות.

שאלה זו רלוונטית למצבים בהם החברה רוצה לראות מיהם הלקוחות החוזרים שלה (עבור מתן צ'ופר לדוגמא)

SELECT Customers.Email, [Full Name] = NameFirst + ' ' + NameLast, [Number of orders] = COUNT(DISTINCT O.OrderID)

FROM Customers JOIN Orders AS O ON Customers.Email = O.Customer

JOIN [Contains] ON O.OrderID = [Contains].OrderID

JOIN Products AS P ON P.DesignID = [Contains].DesignID

GROUP BY Customers.Email, NameFirst, NameLast

HAVING COUNT(DISTINCT O.OrderID) > 3

ORDER BY COUNT(DISTINCT O.OrderID)

תמונה שמכילה טקסט, צילום מסך, גופן, קו

התיאור נוצר באופן אוטומטי

# שתי שאילתות SELECT מקוננות

**שאילתה מספר 1- מקוננת שהקינון מחזיר סקאלר:**

השאילתה מציגה עבור כל שם של עוגה מקטגוריה (עוגה מהקטלוג) את הכמות שנקנתה ממנה ואת אחוז הרכישות של אותה עוגה מבין כל העוגות שמופיעות בקטלוג.

השאלה העסקית: מה הכמות המוזמנת מכל עוגה מהקטלוג?

שאלה זו רלוונטית למצבים בהם החברה רוצה לראות אילו עוגות אהובות יותר/ פחות.

SELECT CakeName, TotalQuantity = SUM(Quantity), QuantityPrecent = SUM(Quantity) \* 100.0 /

(

SELECT SUM(Quantity)

FROM Categories AS CA

JOIN [Contains] AS CO ON CA.DesignID = CO.DesignID

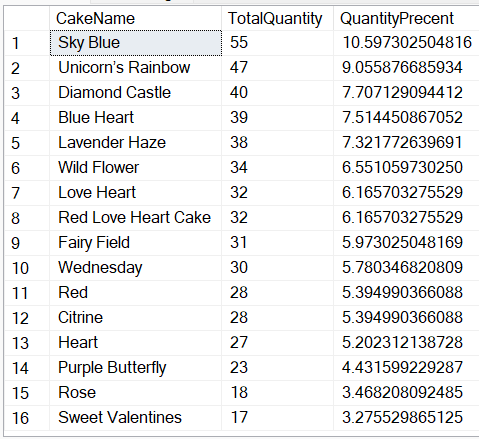
)

FROM Categories AS CA

JOIN [Contains] AS CO ON CA.DesignID = CO.DesignID

GROUP BY CakeName

ORDER BY QuantityPrecent DESC



**שאילתה מספר 2- מקוננת שהקינון מחזיר טבלה:**

השאילתה מציגה את 10 העוגות מהקטלוג (מוכנות מראש) המוזמנות ואת כמות הביקורות שקיבלה כל עוגה.

השאלה העסקית: האם כמות הביקורות משפיעה על פופולריות העוגה?

שאלה זו יכולה להיות רלוונטית לקבלת החלטה האם הביקורות תורמות לעסק או שהן מיותרות\מפחיתות מכירות וניתן להסיר אותן.

SELECT W.Cake, [Total Reviews] = COUNT(W.Review)

FROM (

SELECT TOP 10 CA.DesignID, [Total Quantity] = COUNT(C.Quantity)

FROM Categories AS CA JOIN [Contains] AS C ON CA.DesignID = C.DesignID

GROUP BY CA.DesignID

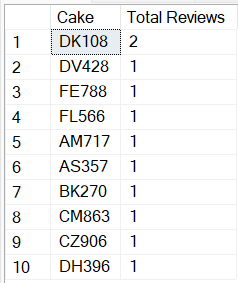
ORDER BY COUNT(C.Quantity) DESC

) AS X

LEFT JOIN Writes AS W ON X.DesignID = W.Cake

GROUP BY W.Cake

# ORDER BY COUNT(W.Review) DESC



# שתי שאילתות מקוננות תוך שימוש במרכיבים נוספים

**שאילתה 1- UPDATE:**

השאילתה מוסיפה לטבלת הלקוחות עמודה המציגה לכל לקוח לאיזה דרגה (Level) הוא שייך לפי כמות ההזמנות שביצע:

-לקוח חדש: הזמנה אחת

-לקוח רגיל: בין 2 ל3 הזמנות

-לקוח VIP : מעל 3 הזמנות   
השאלה העסקית: מהי דרגת הלקוח לפי מספר הרכישות שביצע?  
שאלה זו עוזרת לסווג את הלקוחות ולהבין מי הלקוחות החדשים ומי המתמידים.

ALTER TABLE Customers ADD [Level] VARCHAR(10)

UPDATE Customers SET [Level] = CASE

WHEN (

SELECT COUNT(\*)

FROM Orders O

WHERE O.Customer = Customers.Email

) > 3 THEN 'VIP'

WHEN (

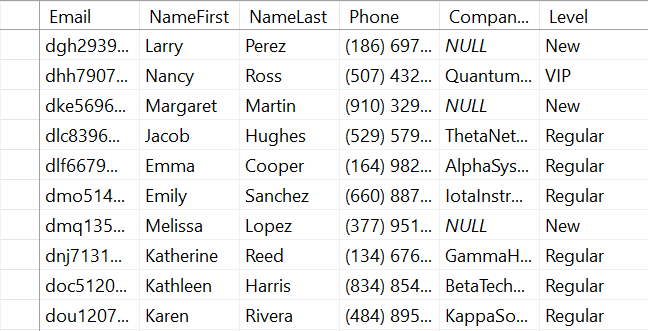
SELECT COUNT(\*)

FROM Orders O

WHERE O.Customer = Customers.Email

) BETWEEN 2 AND 3 THEN 'Regular'

ELSE 'New' END



**שאילתה 2- EXCEPT:**

השאילתה מציגה את הטעמים שלא נמכרו בעיר מלבורן בגודל הפופולארי ביותר.

הרעיון העסקי הוא ניתור של פופולאריות הטעמים בעיר בה ממוקמת החברה.

SELECT DISTINCT Flavor

FROM Products

EXCEPT

SELECT Flavor

FROM Orders AS O JOIN [Contains] AS C ON O.OrderID = C.OrderID

JOIN Products AS P ON C.DesignID = P.DesignID

WHERE AddressCity = 'Melbourne' AND Size =

(

SELECT TOP 1 Size

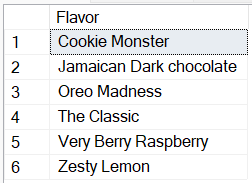
FROM Orders AS O JOIN [Contains] AS C ON O.OrderID = C.OrderID

JOIN Products AS P ON C.DesignID = P.DesignID

GROUP BY Size

ORDER BY COUNT(\*) DESC

)



# שאילתות עסקיות המשלבות Window Functions

**שאילתה 1:**

השאילתה מציגה עבור כל הזמנה את המדינה שאליה בוצעה ההזמנה, את סכום ההזמנה, את דירוג ההזמנה ביחס לשאר ההזמנות במדינה, את ממוצע ההזמנה הכולל(לפי כל המדינות), את ממוצע ההזמנה במדינה, את כמות ההזמנות שבוצעה לכל מדינה ואת כמות ההזמנות הכולל.  
הרעיון העסקי מאחורי השאילתה: לראות מה המצב של כל מדינה ביחס לאחרות (ביחד ולחוד) לפי הממוצע הכללי והממוצע בה, כמות ההזמנות הכללית ובה. לראות מה סכום כל הזמנה ואיפה היא ביחס לשאר ההזמנות במדינה.

ALTER TABLE Orders

ADD [Total Amount] MONEY

UPDATE Orders

SET [Total Amount] = (

SELECT SUM(C.Quantity \* (1 - ISNULL(COUP.DiscountPercent, 0) / 100) \* (P.BasePrice + ISNULL(P.AdditionalPrice, 0)))

FROM [Contains] AS C

JOIN Products AS P ON C.DesignID = P.DesignID

LEFT JOIN Coupons AS COUP ON Orders.CouponName = COUP.CouponName

WHERE Orders.OrderID = C.OrderID

)

SELECT AddressCountry, [Total Amount],

[Order Rank] = RANK() OVER (PARTITION BY AddressCountry ORDER BY [Total Amount] DESC),

[Total AVG] = AVG([Total Amount]) OVER(),

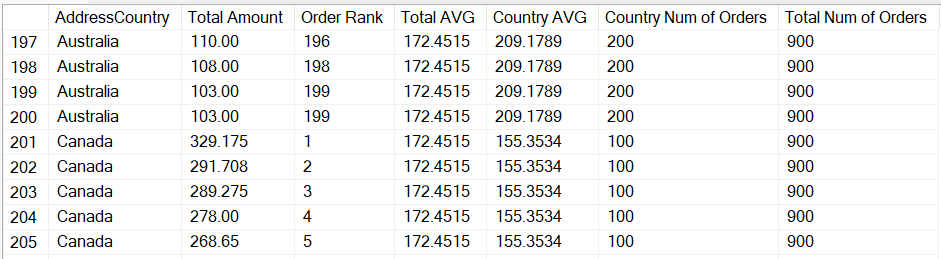
[Country AVG] = AVG([Total Amount]) OVER(PARTITION BY AddressCountry),

[Country Num of Orders] = COUNT([Total Amount]) OVER(PARTITION BY AddressCountry),

[Total Num of Orders] = COUNT([Total Amount]) OVER()

FROM Orders

ORDER BY AddressCountry



**שאילתה 2:**

שאילתה זו מציגה לכל הזמנה מי הלקוח שביצע אותה (Email), מה סכום ההזמנה, כמות ההזמנות שהלקוח ביצע, מה הסכום הכולל של ההזמנות שביצע לקוח זה, ממוצע ההזמנות ללקוח, ממוצע ההזמנות הכולל ובאיזה אחוזון הלקוח נמצא (מתוך 4).  
מטרת השאילתה היא לראות איפה כל לקוח עומד ביחס לשאר והיא מאפשרת לאמוד את תרומתו של כל לקוח.

ALTER TABLE Customers

ADD [Total per Customer] MONEY

UPDATE Customers

SET [Total per Customer] =

(

SELECT SUM([Total Amount])

FROM Orders

WHERE Customers.Email = Orders.Customer

)

SELECT Email, [Total Amount],

[Num of Orders] = COUNT([Total Amount]) OVER(PARTITION BY Email),

[Total per Customer],

[Total AVG] = AVG([Total Amount]) OVER(),

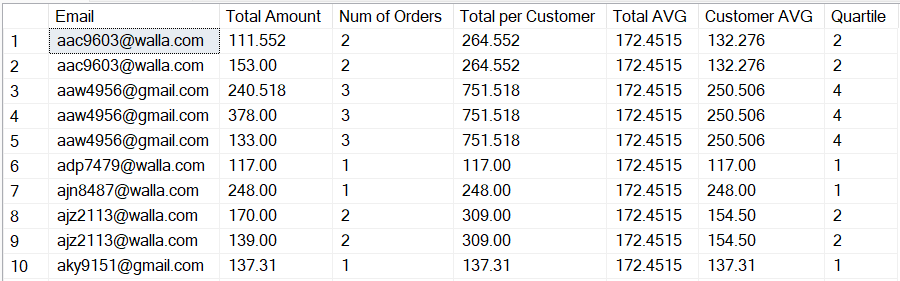
[Customer AVG] = AVG([Total Amount]) OVER(PARTITION BY Email),

Quartile = NTILE(4)

OVER (ORDER BY [Total per Customer])

FROM Customers AS C JOIN Orders AS O ON C.Email = O.Customer

ORDER BY Email



# דו"ח המושתת על שימוש בפסקת WITH מורכבת

ההיגיון מאחורי ה- WITH:

הנהלת החברה בוחנת נתונים לגבי גרסאות הבסיס השונות של העוגה בעיצוב אישי במטרה לבחון אפשרויות של שיפור בסיס מסוים, החלפה לבסיס אחר וכו'. לשם כך נרצה להציג לכל בסיס את הכמות שנקנתה ממנו, את סך ההכנסות מהבסיס, את התרומה היחסית של הבסיס להכנסות, באיזה אחוזון תרומה הוא (מתוך 4), בכמה מדינות שונות נקנה, בכמה ערים שונות נקנה ומה היחס בין מספר הערים שנקנה בו למספר הערים הכולל.

WITH

JoinTable AS (

SELECT O.AddressCountry, O.AddressCity, CU.CakeBase, CO.Quantity, CU.DesignID, O.OrderID,

COUP.CouponName, COUP.DiscountPercent, P.BasePrice, P.AdditionalPrice

FROM Customs AS CU

JOIN Products AS P ON CU.DesignID = P.DesignID

JOIN [Contains] AS CO ON CO.DesignID = P.DesignID

JOIN Orders AS O ON O.OrderID = CO.OrderID

JOIN Coupons AS COUP ON O.CouponName = COUP.CouponName

),

CountCities AS (

SELECT CakeBase,

[Number of Countries] = COUNT(DISTINCT AddressCountry),

[Number of Cities] = COUNT(DISTINCT AddressCity)

FROM JoinTable

GROUP BY CakeBase

),

Price AS (

SELECT CakeBase, [Total Quantity] = SUM(Quantity),

[Revenue from Base] = SUM(Quantity \* (1 - ISNULL(DiscountPercent, 0) / 100) \* (BasePrice + ISNULL(AdditionalPrice, 0)))

FROM JoinTable

GROUP BY CakeBase

),

TotalRevenue AS (

SELECT [Total Revenue] = SUM([Revenue from Base])

FROM Price

),

CountAllCities AS (

SELECT [Count all Cities] = COUNT(DISTINCT AddressCity)

FROM Orders

)

SELECT CC.CakeBase,

P.[Total Quantity],

[Revenue from Base] = ROUND(P.[Revenue from Base],2),

[Revenue Ratio] = ROUND((P.[Revenue from Base] \* 100.0 / TR.[Total Revenue]),2),

[Revenue Quartile] = Ntile(4) OVER (ORDER BY P.[Revenue from Base] DESC),

CC.[Number of Countries],

CC.[Number of Cities],

[Cities Ratio] = CAST((CC.[Number of Cities] \* 100.0 / CAC.[Count all Cities]) AS DECIMAL(10,2))

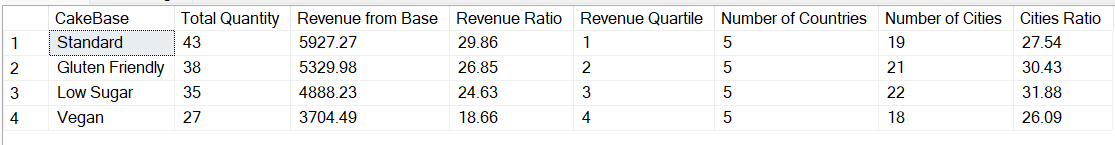
FROM CountCities AS CC

JOIN Price AS P ON CC.CakeBase = P.CakeBase

CROSS JOIN TotalRevenue AS TR

CROSS JOIN CountAllCities AS CAC

ORDER BY P.[Revenue from Base] DESC



## מטלה 2 – כלים להצגת נתונים

**טבלת VIEWS :**

CREATE VIEW View\_RuwisCakes AS

SELECT

o.OrderID,

o.AddressCountry,

o.AddressCity,

o.AddressStreet,

o.AddressHouseNumber,

o.AddressApartment,

c.Email,

c.NameFirst,

c.NameLast,

c.Phone,

c.CompanyName,

p.DesignID,

p.Size,

p.Topper,

p.Flavor,

p.BasePrice,

p.AdditionalPrice,

cu.CakeBase,

cu.CakeIcing,

cu.Topping1,

cu.Topping2,

cu.Topping3,

cat.CakeName,

cat.Color,

con.Quantity,

co.CouponName,

co.DiscountPercent,

cc.CreditCardNumber,

w.Review

FROM Orders AS o

JOIN Customers AS c ON o.Customer = c.Email

JOIN [Contains] AS con ON o.OrderID = con.OrderID

JOIN Products AS p ON con.DesignID = p.DesignID

JOIN CreditCards AS cc ON cc.CreditCardNumber = o.CreditCard

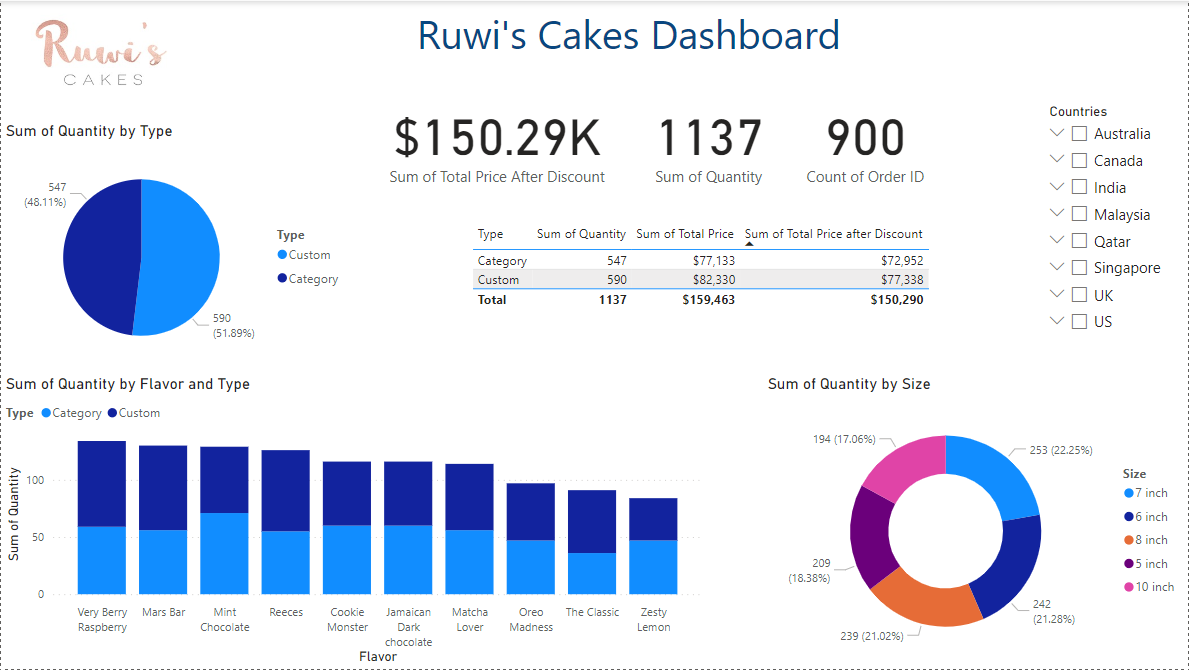
LEFT JOIN Customs AS cu ON p.DesignID = cu.DesignID

LEFT JOIN Categories AS cat ON p.DesignID = cat.DesignID

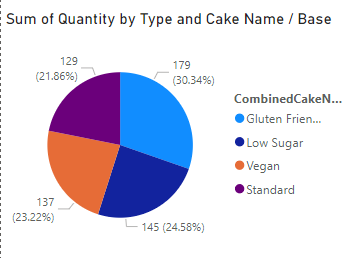
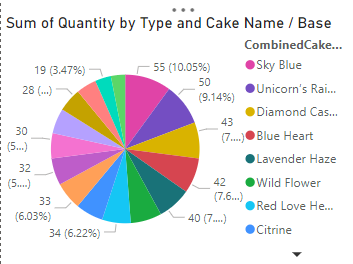
LEFT JOIN Coupons AS co ON o.CouponName = co.CouponName

LEFT JOIN Writes AS w ON w.Cake = cat.DesignID

**לוח מחוונים:**



דיאגרמת העוגה לאחר Drill down לCategory (מימין) ולאחר Custom (משמאל)



**דו"ח עסקי:**