1. <u>מושגים בסיסים:</u> (מהגדול לקטן)

<u>מסד הנתונים</u> – כל שמונת הטבלאות של חברת NorthWind.

<u>טבלה</u> – כל אחת משמונת הטבלאות, הטבלה במבנה של שורות ועמודות. (לדוגמה טבלת הלקוחות, המוצרים וכו')

<u>רשומה</u> – שורה בטבלה שמייצגת פריט יחיד בטבלה, לדוגמה לקוח מתוך כל הלקוחות.

<u>שדה</u> – עמודה בטבלה, לדוגמה שמות הלקוחות, ערי הלקוחות וכו'.

---

2. <u>מבנה הקוד:</u> (המספור זה סדר ההרצה הפנימי בתוכנה)

כתיבת העמודות שאותם נרצה להציג ..... (5) SELECT

הטבלאות מהם נרצה לשלוף את התוכן ..... (1) FROM

התנאים שנרצה לבדוק עבור העמודות ..... (2) WHERE

קיבוץ המידע לקבוצות קטנות עם מכנה משותף למטרת חישוב ..... (3) GROUP BY

התנאים שנרצה לבדוק עבור הקבוצות הקטנות שיצרנו ..... (4) HAVING

מיון מהקטן לגדול (אם נרצה למיין מהגדול לקטן ← DESC)..... (6) ORDER BY

---

3. <u>חידודים לגבי DESC:</u>

ORDER BY Customers.City		DESC
סוג המידע בטבלה	רגיל (ללא DESC)	הפוך (עם DESC)
שדות טקסט	A B C . .	Z Y X . .
מספרים	1 2 3 . .	999 998 997 . .
תאריכים	20.6.2024 21.6.2024 22.6.2024 . .	22.6.2024 21.6.2024 20.6.2024 . .

---

4. <u>שליפת התאריך של היום:</u>

GETDATE() – לדוגמה היום התאריך הוא 22.6.2024.

5. <u>שליפת מידע מהתאריך:</u>

מה רוצים לשלוף	איך זה ייכתב? (לחלק 2 אופציות)	התוצאה
יום בחודש	1. DAY(שדה התאריך) 2. DATEPART(DAY, שדה התאריך)	22
מספר חודש	1. MONTH(שדה התאריך) 2. DATEPART(MONTH, שדה התאריך)	6
שנה	1. YEAR(שדה התאריך) 2. DATEPART(YEAR, שדה התאריך)	2024
יום בשבוע	DATEPART(WEEKDAY, שדה התאריך)	7 (שבת)
רבעון	DATEPART(Quarter, שדה התאריך)	2 (שני)

6. <u>הפרש בין 2 תאריכים:</u>

DATEDIFF(באיזה יחידה ההפרש יוחזר , תאריך קטן , תאריך גדול)

<u>האופציות ליחידות שבהם הפרש התאריכים יוחזר:</u>

DAY, MONTH, YEAR, QUARTER

7. שורת ה-WHERE – עבור מספרים:

א. אפשרות להשתמש באופרטורים לוגיים (>, <, =, =<, =>, <> או =!)
לדוגמה, הצגת ההזמנות שמחיר ההובלה מעל 150:

```
SELECT *
FROM Orders AS O
WHERE O.Freight > 150
```

ב. בטווח ספציפי - BETWEEN
לדוגמה מחירי הובלה בין 20 ל-30:

```
SELECT *
FROM Orders AS O
WHERE O.Freight BETWEEN 20 AND 30
```

(אפשר גם להוסיף NOT לפני המילה BETWEEN ואז זה בעצם כל מה שלא בטווח.)

ג. מתוך קבוצת מספרים – IN:
לדוגמה שמחירי ההובלה, 8 או 9, או 11 או 23:

```
SELECT *
FROM Orders AS O
WHERE O.Freight IN (8, 9, 11, 23)
```

(אפשר להוסיף NOT לפני המילה IN ואז זה בעצם יהיה כל שאר המספרים.)

---

8. שורת ה-WHERE – עבור שדות טקסט:

א. תנאי פשוט: (לדוגמה להחזיר את העובדים עם השם משפחה Zeev)

```
SELECT *
FROM Employees as e
WHERE e.LastName Like 'Zeev'
```

ב. שילוב של 2 תנאים **ביחד**: (לדוגמה גם משפחה Zeev וגם פרטי Ofek) - AND

```
SELECT *
FROM Employees as e
WHERE e.LastName Like 'Zeev' and e.FirstName Like 'Ofek'
```

ג. תנאי ראשון או שני: (לדוגמה שם משפחה Zeev או Yakobi) - OR

```
SELECT *
FROM Employees as e
WHERE e.LastName Like 'Zeev' Or e.LastName Like 'Yakobi'
```

ד. הכל חוץ מספציפי: (לדוגמה כל העובדים חוץ מי ששם המשפחה שלו Ran)

```
SELECT *
FROM Employees as e
WHERE e.LastName NOT Like 'Fuller'
```

9. בדיקת זוגי או אי זוגי באמצעות ה-"**מודולו**" 2%

א. לדוגמה, שליפת ההזמנות שמחיר ההובלה שלהם הוא זוגי: (לפי שארית 0)

```
SELECT *
FROM Orders AS O
WHERE O.Freight % 2 = 0
```

ב. לדוגמה, שליפת ההזמנות שמחיר ההובלה שלהם הוא אי זוגי: (לפי שארית 1)

```
SELECT *
FROM Orders AS O
WHERE O.Freight % 2 = 1
```

ג. מלבד התנאי לזוגי או לא זוגי, אפשרי **לעשות את זה לכל בדיקת חלוקת מספרים ללא שארית**, לדוגמה מתחלקים ב-3, 5, 10 וכו'.

---

10. תבניות טקסט עם % או עם _ (קו תחתון) או עם סוגרים מרובעות [ ] :

לדוגמה כל הלקוחות שהשם שלהם מתחיל באות A, הכיתוב יהיה באופן הבא:

```
SELECT *
FROM Customers AS C
WHERE C.CompanyName LIKE 'A%'
```

א. שימוש ב-% בשביל טקסט, דוגמאות:

טקסט שמתחיל ב-A: 'A%'

טקסט שנגמר ב-A: '%A'

טקסט שמכיל A: '%A%'

טקסט שמכיל את הצירוף OFE: '%OFE%'

טקסט שמתחיל ב-A ונגמר ב-B: 'A%B'

ב. שימוש ב-' _ ' (קו תחתון): (גם להגדרת כמות התווים)

טקסט של 2 תווים בדיוק: '_ _'

טקסט של 2 תווים בדיוק, שמתחיל ב-A: 'A_'

טקסט של 2 תווים בדיוק, שמסתיים ב-A: '_A'

טקסט של 3 תווים בדיוק, שבאמצע A: '_A_'

ג. שימוש גם ב-% וגם בקו תחתון:

טקסט של לפחות 3 תווים: '_ _ _%'

טקסט שהתו השני שלו הוא A: '_A%'

טקסט שמתחיל ב-A, ו-B ספרה לפני הסוף: 'A%B_'

ד. שימוש בסוגרים מרובעים [ ]:

טקסט שמכיל את האות A או B בספרה הראשונה: '%[AB]'

11. <u>החזרת עמודה שכל ערך יופיע רק פעם אחת באמצעות **DISTINCT**</u>: (ללא כפילויות)
לדוגמה אם נרצה מעמודה (שדה) אחת של ערים, שכל עיר תוצג לנו רק פעם אחת,
נשתמש במילה Distinct לפני השדה בשורת ה-SELECT.
לדוגמה, נרצה להציג את המדינות של הלקוחות, ללא כפילויות:

```
SELECT DISTINCT c.Country
FROM Customers AS c
```

---

12. <u>קישור בין טבלאות דרך **INNER JOIN**</u>:
שנרצה לשלב רשומות (שורות) משתי טבלאות **או יותר**, ויש התאמה בערכים של השדות (עמודות) המקשרים (לדוגמה קוד קטגוריה בטבלת הקטגוריות, וקוד קטגוריה בטבלת המוצרים)
נשתמש ב-Inner Join.

כלומר ה-Inner Join מחזיר רק את הרשומות (שורות) מהטבלאות שבהן יש התאמה בערכים בין השדות (עמודות) המשותפים.

לדוגמה, אם נרצה להציג עבור כל המוצרים את שם הקטגוריה שלהם, נבצע את הקישור בשדות CategoryID בטבלת המוצרים ובטלת הקטגורים באופן הבא:

```
SELECT pr.ProductID, ca.CategoryName
FROM Products AS pr INNER JOIN Categories AS ca
    ON pr.CategoryID = ca.CategoryID
```

---

13. <u>קישור בין טבלאות דרך **LEFT JOIN** או **RIGHT JOIN**</u>:
INNER JOIN מסעיף 11, מחזיר **רק את הרשומות שיש להם התאמה** בשתי הטבלאות.
לעומת זאת **LEFT JOIN** מחזיר **את כל הרשומות** מהטבלה השמאלית (הראשונה),
ובהתאם את כל ההתאמות מהטבלה הימנית (השנייה).
ערכים חסרים מהטבלה הימנית יוצגו כ-NULL.

RIGHT JOIN זהה ל-**LEFT JOIN** רק שזה הפוך – הוא יחזיר את כל הרשומות מהטבלה הימנית (השנייה), ובהתאם את כל ההתאמות מהטבלה השמאלית (הראשונה).

לדוגמה, נרצה להציג את פרטי הלקוחות **שלא ביצעו הזמנות**.
לשם כך נרצה להשתמש ב-**LEFT JOIN** שהטבלה השמאלית היא הלקוחות – כי נרצה את כל הלקוחות, והטבלה הימנית היא ההזמנות שיקושרו בהתאם **לכל הלקוחות**.

לאחר שנעשה ביניהם קישור, נעשה תנאי ב-WHERE לבחור את הרשומות (השורות) שקוד ההזמנה שלהם לא קיים (הוא בעצם NULL)

```
SELECT C.*
FROM Customers AS C LEFT JOIN Orders AS O
    ON C.CustomerID = O.CustomerID
WHERE O.OrderID IS NULL
```

14. <u>פונקציות חישוביות:</u>
הפונקציות הבאות משמשות לביצוע חישובים על **עמודה שלמה**, ולהחזיר **ערך מספרי יחיד** שהוא תוצאת החישוב.

א. <u>פונקציית SUM:</u>
באמצעות הפונקציה SUM אפשר לסכום את כל הערכים בעמודה מסוימת.
לדוגמה, הצגת ממוצע מחירי המוצרים:

```
SELECT AVG(pr.UnitPrice)
FROM Products AS pr
```

ב. <u>פונקציית COUNT:</u>
a. <u>שימוש ראשון</u> – **ספירת כל הרשומות** (שורות) בטבלה מסוימת. (בעזרת *)
לדוגמה, ספירת כמות הלקוחות:

```
SELECT COUNT(*)
FROM Customers
```

b. <u>שימוש שני</u> – **ספירת הרשומות ששדה מסוים בהם לא ריק**.
לדוגמה, ספירת כל הלקוחות שיש להם מחוז (Region):

```
SELECT COUNT(cu.Region)
FROM Customers AS cu
```

c. <u>שימוש שלישי</u> – ספירת **השדות הייחודים**, כלומר כמה ערכים שונים יש בעמודה מסוימת? (שימוש במילה Distinct)
לדוגמה, ספירת כל החזות השונים הקיימים:

```
SELECT COUNT (DISTINCT cu.Region)
FROM Customers AS cu
```

ג. <u>פונקציית MAX:</u>
באמצעות הפונקציה MAX נחזיר מעמודה מסוימת את הערך המספרי הכי גדול.
לדוגמה, מכל ההזמנות נרצה להחזיר את העלות הובלה הכי גבוהה שהייתה:

```
SELECT MAX (o.ShipVia)
FROM Orders AS o
```

ד. <u>פונקציית MIN:</u>
באמצעות הפונקציה MIN נחזיר מעמודה מסוימת את הערך המספרי הכי קטן.
לדוגמה, מכל ההזמנות נרצה להחזיר את העלות הובלה הכי נמוכה שהייתה:

```
SELECT MIN (o.ShipVia)
FROM Orders AS o
```

ה. <u>פונקציית AVG:</u>
באמצעות הפונקציה AVG נחזיר את הממוצע המספרי של עמודה מסוימת.
לדוגמה, מכל ההזמנות נרצה להחזיר את ממוצע ההובלות:

```
SELECT AVG (o.ShipVia)
FROM Orders AS o
```

15. <u>פסוקית GROUP BY:</u>

פסוקית Group By מאפשרת לנו לקבץ את תוצאות השאילתות לפי קבוצות מוגדרות.
תמיד נדרש לקבץ (גם) לפי מפתח!

מה שיקרה בקוד:
1. קודם כל יפצל לנו את הטבלה **למיני טבלאות** לפי הקטגוריות שבחרנו.
2. הקוד יבוצע בנפרד על כל **מיני טבלה**.
3. כל המידע מכל **המיני טבלאות** יכניסו לטבלה הגדולה שלנו, בשורות נפרדות.

<u>לדוגמה, נרצה להציג עבור כל ספק:</u>
כמה מוצרים הוא מספק, ממוצע מחירי המחירון של מוצריו, והמחיר המקסימלי שלהם:

```
SELECT S.CompanyName,
       COUNT(*) AS CountSuppProducts,
       AVG(P.UnitPrice) AS 'Avg Price',
       MAX(P.UnitPrice) AS 'Max Price'
FROM Suppliers AS S INNER JOIN Products AS P
     ON P.SupplierID = S.SupplierID
GROUP BY S.SupplierID, S.CompanyName
```

<u>בדוגמה הנוכחית:</u>
1. ראשית טבלת הספקים אוחדה עם טבלת המוצרים.
2. לאחר מכן הקוד פוצל **למיני טבלאות** לפי ספקים (GROUP BY S.SupplierID)
3. ואז עבור **כל מיני טבלה** נבחרו (הוצגו) השדות הבאים בלבד:
   שם הספק, כמות השורות (שזה בעצם המוצרים שהספק מספק), מחיר ממוצע של מחיר המחירון, והמחיר מחירון הכי גבוה.
4. לבסוף כל **המיני טבלאות** מתאחדים לנו לטבלה אחת שמסכמת את כל הספקים.

---

16. <u>פסוקית HAVING:</u>

פסוקית HAVING היא פקודה אשר משמשת לסנן קבוצות נתונים שנוצרו על ידי פקודת GROUP BY.

<u>ההבדל בין HAVING ל-WHERE:</u>
WHERE עושה סינון **לרשומות** בודדות **לפני** הקיבוץ נתונים.
HAVING עושה סינון **לקבוצות** נתונים **לאחר** הקיבוץ נתונים.

בהמשך לדוגמה מההסבר על GROUP BY, כעת נדרש להציג רק את הספקים שהמחיר המקסימלי שלהם גדול מ-40:

```
SELECT S.CompanyName,
       COUNT(*) AS CountSuppProducts,
       AVG(P.UnitPrice) AS 'Avg Price',
       MAX(P.UnitPrice) AS 'Max Price'
FROM Suppliers AS S INNER JOIN Products AS P
     ON P.SupplierID = S.SupplierID
GROUP BY S.SupplierID, S.CompanyName
HAVING MAX(P.UnitPrice) > 40
```

17. <u>שדות חסרות ערך – NULL:</u>
כאשר נרצה לדעת עבור שדה (עמודה) האם קיים בו ערכים או לא, יש לו 2 אופציות:
1. **<u>is Null</u> – משמש בבדיקה** עבור שדות **ללא ערך**.
2. **<u>Is Not Null</u> – משמש בבדיקה** עבור שדות עם ערך **בלבד**.

<u>לדוגמה, נרצה להציג את רשימת הלקוחות **בעלי הפקס**:</u>

```
SELECT *
FROM Customers AS C
WHERE C.Fax IS NOT NULL
```

<u>דוגמה נוספת, נציג את רשימת הלקוחות **שאין להם פקס**:</u>

```
SELECT *
FROM Customers AS C
WHERE C.Fax IS NULL
```

---

18. <u>הצגת נתונים מכמה שאילתות יחד דרך איחוד של כמה שאילתות – **UNION**:</u>
הפקודה Union משמשת לאיחוד תוצאות של 2 שאילתות נפרדות לטבלה אחת.
הנתונים בשתי השאילתות **חייבים להיות בעלי אותו מבנה**:
אותו מספר עמודות, באותו הסדר, ואותו סוג נתונים בכל עמודה.

בדרך כלל נשתמש שנרצה לאחד נתונים משתי טבלאות דומות שיש להם אותו מבנה.
לדוגמה, שמות לקוחות ומספרי הטלפון שלהם, ושמות עובדים ומספרי הטלפון שלהם.

הכותרות של הטבלה המאוחדת יגיעו מה-SELECT הראשון.
במידה ונרצה למיין את כלל הטבלה, נעשה את המיון תחת השאילתה האחרונה.
אם בשאילתה מסוימת חסר שדה נכתוב ' ' או NULL ואז יהיה לנו עמודה (שדה) ריקה אבל הכמות שדות (עמודות) יהיו זהות.

לדוגמה, נרצה להציג עבור ברזיל את הלקוחות ואת הספקים שגרים שם **בטבלה אחת**.
עבור כל אחד שיהיה רשום לקוח או ספק, שם החברה, שם האיש קשר, ומספר הטלפון.
ולבסוף **למיין את כל הרשימה לפי שם הלקוח / הספק**:

```
SELECT 'CUST' AS INFO,
       C.CompanyName,
       C.ContactName,
       C.Phone
FROM Customers AS C
WHERE C.Country LIKE 'BRAZIL'
UNION
SELECT 'SUPP',
       S.CompanyName,
       S.ContactName,
       S.Phone
FROM Suppliers AS S
WHERE S.Country LIKE 'BRAZIL'
ORDER BY C.ContactName
```

19. <u>הצגת קבועים ב-SQL:</u>

הצגת קבועים ב-SQL היא דרך פשוטה להציג ערכים קבועים בתוצאות של שאילתות.

<u>לדוגמה ניתן ליצור שאילתה עם תא אחד, שהתוכן בו הוא Hello World:</u>

```
SELECT 'Hello World'
```

<u>כמו כן, ניתן ליצור שאילתה שתוכן התא הוא Hello World אבל הכותרת תהייה Ofek:</u>

```
SELECT 'Hello World' AS 'Ofek'
```

---

20. <u>תת שאילתה ב-WHERE:</u> (עבור תנאי בודד)

במידה ונרצה ב-WHERE לבצע השוואה לנתון אחר שעדיין לא חושב לנו, נוכל ליצור תת שאילתה שתחזיר לנו את הנתון, ואז ישירות נבצע את ההשוואה מולו.

לדוגמה, נרצה להציג את שמות כל המוצרים שהמחיר מחירון שלהם (UnitPrice) מתחת לממוצע מחירי המחירון.

לשם כך נחשב בתת שאילתה את ממוצע מחירי המחירון, ובשאילתה הראשית נעשה את כל השאר:

```
SELECT pr.ProductName
FROM Products AS pr
WHERE pr.UnitPrice < (SELECT AVG(pr.UnitPrice)
                      FROM Products AS pr)
```

---

21. <u>תת שאילתה ב-WHERE:</u> (עבור תוצאות מרובות)

במידה ונרצה ב-WHERE לבצע השוואה **אך הפעם למספר נתונים** שעדיין לא חושבו, נצטרך ליצור תת שאילתה שתחזיר לנו **רשימת ערכים**, ואז נבצע את ההשוואות ישירות לתת שאילה הזאת. (כלומר נעשה שימוש ב-**IN** או ב-**NOT IN**)

לדוגמה, נרצה להציג את כל שמות העובדים שטיפלו בהזמנות שהיו בהם מוצרים מקטגוריית משקאות (Beverages). לכן בתת שאילתה נחזיר את כל קודי המוצרים מקטגוריית המשקאות, נעשה שימוש ב-**IN**, ואת כל השאר נעשה בשאילתה הראשית:

```
SELECT DISTINCT em.FirstName, em.LastName
FROM Employees AS em INNER JOIN Orders AS o
        ON em.EmployeeID = o.EmployeeID INNER JOIN [Order Details] AS OD
        ON O.OrderID = OD.OrderID INNER JOIN Products AS P
        ON OD.ProductID = P.ProductID
WHERE  P.ProductID IN (SELECT P.ProductID
                       FROM Products AS P INNER JOIN Categories AS C
                       ON P.CategoryID = C.CategoryID
                       WHERE C.CategoryName LIKE 'Beverages')
```

22. <u>תת שאילתה ב-HAVING:</u> **(עבור תנאי בודד בלבד)**
בדומה לתת שאילתה ב-WHERE, התת שאילתה ב-HAVING עובד על אותו עיקרון.
ההבדל הוא שהתת שאילתה ב-HAVING נועדה להשוואה **עבור הסינון ברמת הקבוצות**.
כאן אין אופציה לעשות **IN** או **NOT IN**, אלא רק השוואה לתוצאה בודדת.

לדוגמה, נרצה להציג את שמות הקטגוריות שמחיר המחירון הממוצע של מוצריהן גבוה ממחיר המחירון הממוצע של המוצרים בקטגורית Produce:

```
SELECT C.CategoryName
FROM Categories AS C INNER JOIN Products AS P
        ON P.CategoryID = C.CategoryID
GROUP BY C.CategoryID, C.CategoryName
HAVING AVG(P.UnitPrice) > (SELECT AVG(P.UnitPrice)
                           FROM Products AS P INNER JOIN Categories AS C
                                ON C.CategoryID = P.CategoryID
                           WHERE C.CategoryName = 'PRODUCE')
```

---

23. <u>תת שאילתה ב-SELECT:</u>
תת שאילתה ב-SELECT היא שאילתה המבוצעת בתוך שאילתת SELECT אחרת. בדיוק כמו שתת שאילתה ב-WHERE או ב-HAVING מסיעת בסינון נתונים, גם התת שאילתה ב-SELECT נועדה להחזיר ערכים המוצגים בתוצאה בתוך השאילתה הראשית.
ניתן להחזיר נתונים שיכולים להיות מוצגים בעמודה אחת או יותר של התוצאה הראשית.
התת שאילתה יכולה להחזיר ערך בודד או קבוצת ערכים, בהתאם לצורך.

לדוגמה, נרצה להציג לכל לקוח את סכום עלויות ההובלה **בכל** ההזמנות שלו, ואת ההפרש בין סכום כל עלויות ההובלה בהזמנות של כל הלקוחות לסכום ההובלות שלו.
חישוב סה"כ עלויות ההובלה של כל הלקוחות תחושב באמצעות תת שאילתה ב-SELECT.

```
SELECT C.CustomerID, C.CompanyName,
        SUM(O.Freight) AS 'CustomerSumFreight',
         (SELECT SUM(O. Freight)
          FROM Orders AS O) – SUM(O. Freight) AS 'Different'
FROM Customers AS C INNER JOIN Orders AS O
        ON C.CustomerID = O.CustomerID
GROUP BY C.CustomerID, C.CompanyName
```

---

24. <u>טיפים נוספים:</u>

א. <u>חישוב פדיון</u> - המחיר שנמכרו כל היחידות ממוצר ספציפי בהזמנה ספציפית, יחושב: OD.UnitPrice כפול OD.Quantity.
במידה ונצטרך לחשב את סך הפדיון בהזמנות נעשה: SUM(OD.Quantity * OD.UnitPrice)

ב. <u>שם לקוח</u> – Customers.CompanyName | <u>שם ספק</u> – Suppliers.CompanyName

ג. <u>הקישור היחיד עם שמות לא זהים:</u> (טבלת Orders לטבלת Shippers)
יתבצע דרך השדה **ShipVia בטבלת Orders** לבין השדה **ShipperID בטבלת Shippers**.

ד. <u>דוגמה לשימוש IN ב-WHERE עבור טקסט:</u> WHERE C.Country IN ('USA', 'UK', 'FRANCE')