


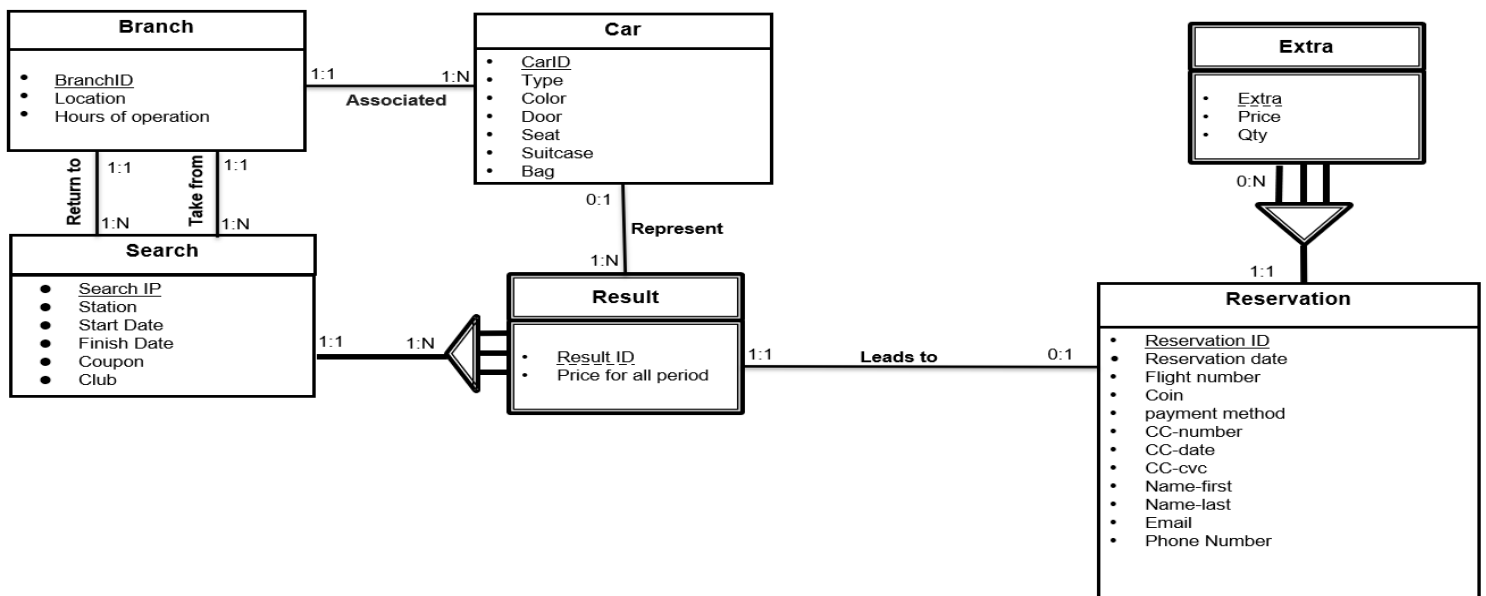
פרויקט בסיסי נתונים חלק ב'	אוניברסיטת בן גוריון בנגב	
	המחלקה להנדסת תעשייה וניהול	

מס' קבוצה	אתר	תאריך הגשה
20	השכרת רכב בחו"ל	26.01.24
חברי הצוות - מספרי ת.ז.		
322816885	207476151	206364911

חלק ג'-פרק ראשון – מטלות חובה

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

ה-ERD ומבנה הטבלאות בהגשת חלק ב':



- **BRANCHES** (BranchID, Location, HoursOfOperation)
- **CARS** (CarID, Car Type, Gear,Color, Door, Seat, Suitcase, Bag, BranchID(BRANCHES))
- **SEARCHES** (Search IP, Station, Start Date, Finish Date, Club,_BranchT (BRANCHES), BranchR(BRANCHES))
- **RESULTS**(SearchID(SEARCHES), ResultID, Price for all period, CarID(CARS)))
- **CARDS**(CC-number, CC-date, CC-cvc)
- **RESERVATIONS** (ReservationID, Reservation date, Flight number, Coin, Payment-method, Name-first, Name-last, Email, Phone Number,CC-number(CARDS), CarID

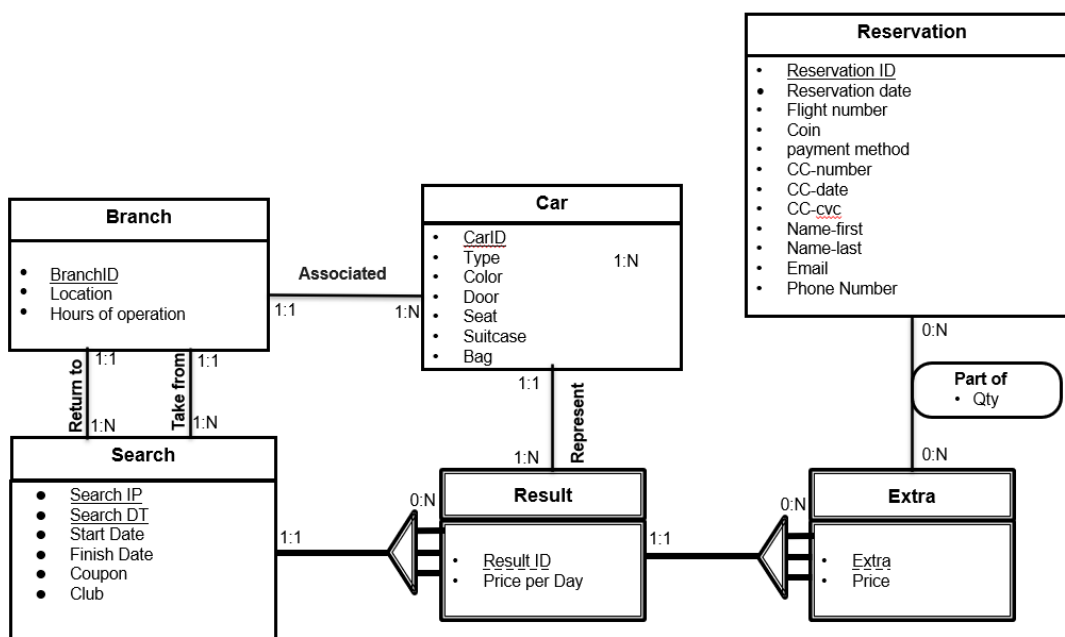


(CARS))

{ERSSearchID, ResultID}(RESULTS))

- **EXTRAS** (Reservation ID(RESERVATIONS), Extra, Price, Qty)

תרשים ERD-ה המתוקן עליהם הסתמכנו בחלק זה של העבודה:



תיאור מילולי לתיקונים:

1. מאחר ויכולים לבצע מספר חיפושים מאותו המחשב (אותה כתובת IP) הוספנו לישות Search מזהה נוסף Search DT ההופך את המזהה של הישות לייחודי, במחשב מסוים יכול להתבצע בדיוק חיפוש אחד בתאריך ושעה מסוימים.
2. הורדנו את השדה Station מהישות Search מאחר והמשמעות שלו מתבטאת בקשרים בין Search ל- Branch.
3. שינינו את הקרדינליות בין Search ל- Result (מ-0:1 ל-1:1) מהסיבה שאם ביצענו חיפוש ולא הופיעה אף מכונת - פשוט לא יהיו לי מופעים של הישות Result, ולכן תוצאה אחת או יותר מייצגת מכונת אחת בדיוק.
4. שינינו את הישות אקסטרה להיות ישות חלשה של Result כיוון ש Result מייצג לנו תוצאה של מכונת מסוימת ותוספות לרכב זמינות פר מכונת, בנוסף קישרנו את אקסטרה לישות Reservation והורדנו את השדה QTY מאקסטרה להיות תכונה של הקשר בין הישויות.

פרויקט בסיסי נתונים חלק ב'	אוניברסיטת בן גוריון בנגב	
	המחלקה להנדסת תעשייה וניהול	

מבנה הטבלאות המתוקן:

- **BRANCHES** (BranchID, Location, HoursOfOperation)
- **CARS** (CarID, Gear, Color, BranchID(BRANCHES), Car Type(CAR TYPES))
- **CAR TYPES** (Car Type, Door, Seat, Suitcase, Bag)
- **SEARCHES** (Search IP, Search DT, Start Date, Finish Date, Club, BranchT (BRANCHES), BranchR(BRANCHES))
- **RESULTS** ({SearchIP,SearchDT} (SEARCHES), ResultID, Price per day , CarID(CARS))
- **CARDS** (CC-number, CC-date, CC-cvc)
- **EXTRAS** ({SearchID,SearchDT,ResultID}(RESULTS), Extra, Price)
- **RESERVATIONS** (ReservationID, Reservation date, Flight number, Coin, Payment-method, Name-first, Name-last, Email, Phone Number, CC-number(CARDS))
- **EXTRASOFRESERVATION** ({SearchID,SearchDT,ResultID,Extra}(EXTRAS), ReservationID)(RESERVATIONS) , Qty)

הנחות נוספות:

1. בטבלה reservations בשדה club הערך הוא 0 עבור לקוחות במחלקת "Private", ו-1 עבור "Business".
2. השדה Extra והשדה Price בישות Extra יכולים לקבל ערכים של 'NO EXTRA' ו-0 בהתאמה.

פרויקט בסיסי נתונים חלק ב'	אוניברסיטת בן גוריון בנגב המחלקה להנדסת תעשייה וניהול
-------------------------------	--



מטלה 1 (35%) – שאילתות

שתי שאילתות **SELECT** ללא קינון (5%, 2.5% לכל שאילתה)

1. הצג את פרטי החיפוש של חיפושים אשר לא הובילו להזמנה ? יש להציג בסדר יורד לפי זמן החיפוש

• הגיון עסקי:

ברצון החברה להשתפר ולהתפתח בצורה שתוביל לעלייה במכירות שלה, על מנת כך עליה לחקור בהמשך את הסיבות שבגללן היו חיפושים שלא הובילו להזמנות בהמשך, האם תוצאות החיפוש לא התאימו לצרכיהם או בכלל לא היו קימות, האם המחיר היה גבוה מידי?, האם היו חסרות תוספות מסוימות לרכב ? על מנת לענות על השאלה שלב מקדים הוא לקבץ את החיפושים הללו.

```
SELECT S.SearchDT,S.SearchIP,S.StartDate,S.FinishDate,S.BranchT,S.BranchR,R.ResultID,
E.Extra
FROM SEARCHES AS S LEFT JOIN RESULTS AS R ON S.SearchIP = R.SearchIP
AND S.SearchDT = R.SearchDT LEFT JOIN
EXTRAS AS E ON R.SearchIP = E.SearchIP
AND R.SearchDT = E.SearchDT
AND R.ResultID = E.ResultID LEFT JOIN
EXTRASOFRESERVATION AS ER ON E.SearchIP = ER.SearchIP
AND E.SearchDT = ER.SearchDT
AND E.ResultID = ER.ResultID
AND E.Extra = ER.Extra
WHERE ER.ReservationID IS NULL
ORDER BY S.SearchDT DESC
```



פלט השאילתה:

1026 רשומות

	SearchDT	SearchIP	StartDate	FinishDate	BranchT	BranchR	ResultID	Extra
1	2025-01-16 19:50:00.0000000	68.206.201.85	2025-02-09	2025-02-21	163	140	NULL	NULL
2	2025-01-14 11:00:00.0000000	130.167.81.81	2025-02-09	2025-03-14	234	333	6854192	No Extra
3	2025-01-13 08:43:00.0000000	135.105.53.182	2025-02-05	2025-03-22	878	386	4897561	First Aid Kit
4	2025-01-12 11:38:00.0000000	13.212.35.36	2025-02-09	2025-03-01	353	18	3151262	Baby Seat
5	2025-01-10 21:08:00.0000000	89.223.252.105	2025-01-23	2025-03-01	402	561	NULL	NULL
6	2025-01-09 03:58:00.0000000	13.83.224.31	2025-01-16	2025-03-02	909	957	4776798	WiFi
7	2025-01-08 20:55:00.0000000	207.165.237.107	2025-01-13	2025-01-21	698	75	NULL	NULL
8	2025-01-07 14:06:00.0000000	208.31.127.222	2025-01-08	2025-02-08	951	516	NULL	NULL
9	2025-01-07 05:12:00.0000000	52.86.153.166	2025-01-18	2025-02-04	801	206	NULL	NULL
10	2025-01-06 05:30:00.0000000	213.137.238.123	2025-02-04	2025-03-16	109	97	NULL	NULL
11	2025-01-02 07:15:00.0000000	56.181.89.139	2025-01-08	2025-01-29	135	923	NULL	NULL
12	2025-01-02 04:49:00.0000000	13.193.43.171	2025-01-19	2025-02-11	764	522	7328650	WiFi
13	2025-01-01 13:56:00.0000000	39.162.219.43	2025-01-17	2025-02-08	249	484	NULL	NULL
14	2024-12-29 02:42:00.0000000	218.83.163.181	2025-01-28	2025-02-28	858	823	NULL	NULL
15	2024-12-28 08:53:00.0000000	123.238.9.226	2025-01-11	2025-02-14	369	282	3316007	No Extra
16	2024-12-25 23:43:00.0000000	128.28.57.132	2025-01-04	2025-01-31	60	898	7051048	WiFi
17	2024-12-22 12:22:00.0000000	90.22.254.16	2025-01-02	2025-01-20	844	121	NULL	NULL

2. שאילתה עם מרכיבי GROUP BY ו- HAVING ופונקציית המערכת:
מהי כמות ההזמנות הכוללת בסדר יורד עבור כל סוג רכב המוחזק באויס שמחיר
השכרתו עולה על 1000 וכמות הזמנות עולה על 2 ?

הגיון עסקי:


החברה רוצה לדעת איזה רכבים מניבים להם הכנסה גבוהה יחסית ובכך לשער איזה רכבים
כדאי לה לקנות.

```
SELECT C.CarType,R.PricePerDay,
       TOTAL_RENT=COUNT(RS.RESERVATIONID)
FROM CARS AS C JOIN RESULTS AS R ON C.CARID=R.CARID
       JOIN EXTRASOFRESERVATION AS E ON E.RESULTID=R.RESULTID
       JOIN RESERVATIONS AS RS ON RS.RESERVATIONID=E.RESERVATIONID
GROUP BY C.CARTYPE,R.PricePerDay
HAVING COUNT(RS.RESERVATIONID) >= 2 AND R.PricePerDay > 1000
ORDER BY TOTAL_RENT DESC
```



• פלט השאילתה:

	CarType	PricePerDay	TOTAL_RENT
1	Kia Sorento	5455.00	8
2	Subaru Forester	2155.00	4
3	Ford Mustang	5558.00	2
4	Hyundai Santa Fe	5855.00	2
5	Ford Mustang	22848.00	2
6	Honda CR-V	48488.00	2
7	Subaru Impreza	55852.00	2
8	BMW X5	66432.00	2
9	Hyundai Tucson	447829.00	2
10	Kia Optima	5985689.00	2

פרויקט בסיסי נתונים חלק ב'	אוניברסיטת בן גוריון בנגב	
	המחלקה להנדסת תעשייה וניהול	

שתי שאילות **SELECT** מקוננות (10%, 5% לכל שאילתה)

1. שאילתה מקוננת המקבלת ערך סקלרי (קיוון ב-**having**):הצג בסדר יורד את הסניפים שממוצע המכירות שלהם נמוך מממוצע המכירות הכללי ואת ממוצע המכירות

• הגיון עסקי:

השאילתה מזהה סניפים שהביצועים שלהם (במונחי ממוצע הכנסות ממכירות) נמוכים מהממוצע הכולל של כל הסניפים. כך ניתן להתמקד בסניפים החלשים כדי לשפר את ביצועיהם.

```

select c.branchID,
year_avg_income=avg(RE.pricePerDay*
DATEDIFF(DAY,S.StartDate,S.FinishDate)+EXR.Qty*E.Price)
from reservations as R join extrasofreservation
as EXR on EXR.reservationID=R.reservationID
join RESULTS AS RE ON RE.ResultID=EXR.ResultID
join cars as c on c.CarID=re.CarID
join searches as S on S.SearchIP=RE.SearchIP AND S.SearchDT=RE.SearchDT
join EXTRAS as E on E.SearchIP=S.SearchIP AND S.SearchDT=E.SearchDT AND E.ResultID =
RE.ResultID
group by C.BranchID
having avg(RE.pricePerDay* DATEDIFF(DAY,S.StartDate,S.FinishDate)+EXR.Qty*E.Price)<
(
select avg(RE.pricePerDay* DATEDIFF(DAY,S.StartDate,S.FinishDate)+EXR.Qty*E.Price)
from reservations as R join extrasofreservation
as EXR on EXR.reservationID=R.reservationID
join RESULTS AS RE ON RE.ResultID=EXR.ResultID
join cars as c on c.CarID=re.CarID
join searches as S on S.SearchIP=RE.SearchIP AND S.SearchDT=RE.SearchDT
join EXTRAS as E on E.SearchIP=S.SearchIP and S.SearchDT=E.SearchDT AND E.ResultID =
RE.ResultID
)
order by year_avg_income desc

```

פרויקט בסיסי נתונים חלק ב'	אוניברסיטת בן גוריון בנגב	
	המחלקה להנדסת תעשייה וניהול	

● פלט השאילתה:

667 רשומות

	branchID	year_avg_income
1	435	1421972.170000
2	833	1305235.211428
3	129	1019198.170000
4	778	925655.820000
5	883	906373.846428
6	12	868489.771428
7	570	717536.240000
8	127	682078.800000
9	73	681912.000000
10	896	681456.600000
11	132	675787.588750
12	4	666804.000000
13	921	652700.075000
14	535	594048.000000
15	538	593642.680000
16	847	568966.800000
17	827	565076.785000
18	529	538698.516000
19	323	536975.370000
20	225	528431.990000
21	919	512809.950000
22	562	509209.500000
23	188	496020.300000
24	973	422370.000000
25	801	400658.424000
26	831	395039.480000

פרויקט בסיסי נתונים חלק ב'	אוניברסיטת בן גוריון בנגב המחלקה להנדסת תעשייה וניהול
-------------------------------	--



2. שאילתה מקוננת המקבלת גם ערך סקלרי וגם טבלאי (קינון ב-**having** וב-**from**):
הצג את כל התוספות הנמכרות ביותר על פי סוג מכונית.

• הגיון עסקי:

מטרת השאילתה היא לזהות את התוספות הפופולריות ביותר לכל סוג רכב, על מנת למקסם רווחים, לייעל את ניהול המלאי, לשפר את אסטרטגיית השיווק, ולהתאים את חוויית הלקוח לצרכים המדויקים שלו.

```

SELECT CarType, COUNT_EXTRA.Extra, MaxQty= MAX(COUNT_EXTRA.CountExtra)
FROM (
  SELECT
    C.CarType,
    ER.Extra,
    CountExtra=SUM(ER.Qty)
  FROM CARS AS C
  JOIN RESULTS AS R ON C.CarID = R.CarID
  JOIN EXTRAS AS E ON R.ResultID = E.ResultID
  JOIN EXTRASOFRESERVATION AS ER ON
    E.SearchIP = ER.SearchIP AND
    E.SearchDT = ER.SearchDT AND
    E.ResultID = ER.ResultID AND
    E.Extra = ER.Extra
  GROUP BY C.CarType, ER.Extra
) AS COUNT_EXTRA

WHERE COUNT_EXTRA.CountExtra = (
  SELECT MAX(Max_Extra.CountExtra)
  FROM (
    SELECT
      C2.CarType,
      ER2.Extra,
      CountExtra=SUM(ER2.Qty)
    FROM CARS AS C2
    JOIN RESULTS AS R2 ON C2.CarID = R2.CarID
    JOIN EXTRAS AS E2 ON R2.ResultID = E2.ResultID
    JOIN EXTRASOFRESERVATION AS ER2 ON
      E2.SearchIP = ER2.SearchIP AND
      E2.SearchDT = ER2.SearchDT AND
      E2.ResultID = ER2.ResultID AND
      E2.Extra = ER2.Extra
    GROUP BY C2.CarType, ER2.Extra
  ) AS Max_Extra
  WHERE Max_Extra.CarType = COUNT_EXTRA.CarType
)

```

פרויקט בסיסי נתונים חלק ב'	אוניברסיטת בן גוריון בנגב	
	המחלקה להנדסת תעשייה וניהול	

GROUP BY COUNT_EXTRA.CarType, COUNT_EXTRA.Extra, COUNT_EXTRA.CountExtra
order by COUNT_EXTRA.CountExtra desc

• פלט השאילתה:
31 רשומות

	CarType	Extra	MaxQty
1	Mercedes GLE	Baby Seat	77
2	Kia Sorento	GPS	75
3	Audi A4	Baby Seat	75
4	BMW X5	Prepaid Fuel	75
5	Mercedes C-Class	Insurance	70
6	Ford Focus	Insurance	70
7	Ford Mustang	GPS	69
8	Hyundai Elantra	WiFi	66
9	Subaru Forester	Baby Seat	66
10	BMW 3 Series	WiFi	64
11	Mazda CX-5	GPS	64
12	Hyundai Tucson	Insurance	63
13	Honda CR-V	Baby Seat	60
14	Toyota Camry	Prepaid Fuel	56
15	Hyundai Santa Fe	Ski Rack	56
16	BMW X3	Baby Seat	56
17	Toyota Corolla	GPS	55
18	Audi A6	WiFi	52
19	Audi Q5	Baby Seat	50
20	Subaru Impreza	WiFi	50
21	Kia Optima	GPS	49
22	Mazda 6	GPS	48
23	Subaru Outback	WiFi	47
24	Kia Sportage	Baby Seat	47
25	Ford Explorer	WiFi	43
26	Mazda 3	WiFi	42
27	Mercedes E-Class	GPS	42
28	Mercedes E-Class	First Aid Kit	42
29	Honda Civic	Prepaid Fuel	41
30	Honda Accord	Baby Seat	38
31	Toyota RAV4	WiFi	35

פרויקט בסיסי נתונים חלק ב'	אוניברסיטת בן גוריון בנגב המחלקה להנדסת תעשייה וניהול
-------------------------------	--



שאלות עסקיות המשלבות Window Functions (10%, 2.5% לכל שימוש נכון בפונקציה)

1. שאלת המשלבת פונקציית חלון (שימוש ב **sum()**, **rank()**):
לכל סניף ולכל שנת פעילות מה סכום המכירות הכולל עבור אותה שנה, מה סכום המכירות המצטבר לאורך השנים עבור הסניף, ומה המדרג של השנה לעומת השנים האחרות בהתייחס למכירות שבוצעו על ידי הסניף. ההיגיון העסקי מאחורי שאלתה זו היא לבחון איזה סניפים הינם הרווחים ביותר, לראות היכן המכירות ירדו ולהבין את הסיבה לירידה.

```
SELECT
    C.BranchID,
    Year_Of_Reservation=YEAR(RES.ReservationDate) ,
    Yearly_Amount=SUM(R.PricePerDay * DATEDIFF(DAY, S.StartDate, S.FinishDate) + EXR.Qty * E.Price),
    Cumulative_Amount=SUM(SUM(R.PricePerDay * DATEDIFF(DAY, S.StartDate, S.FinishDate) + EXR.Qty *
E.Price))
    OVER (PARTITION BY C.BranchID ORDER BY YEAR(RES.ReservationDate)),
    Year_Rank=RANK() OVER (PARTITION BY C.BranchID ORDER BY SUM(R.PricePerDay * DATEDIFF(DAY,
S.StartDate, S.FinishDate) + EXR.Qty * E.Price) DESC)
FROM
    SEARCHES AS S
JOIN
    RESULTS AS R
    ON S.SearchIP = R.SearchIP AND S.SearchDT = R.SearchDT
JOIN
    CARS AS C
    ON C.CarID = R.CarID
JOIN
    EXTRAS AS E
    ON E.SearchIP = S.SearchIP AND E.SearchDT = S.SearchDT
JOIN
    EXTRASOFRESERVATION AS EXR
    ON E.ResultID = EXR.ResultID AND E.Extra = EXR.Extra
    AND E.SearchIP = EXR.SearchIP AND E.SearchDT = EXR.SearchDT
JOIN
    RESERVATIONS AS RES
    ON EXR.ReservationID = RES.ReservationID
GROUP BY
    C.BranchID, YEAR(RES.ReservationDate)
ORDER BY
    Year_Of_Reservation DESC;
```



• פלט השאילתה:
1134 רשומות

	BranchID	Year_Of_Reservation	Yearly_Amount	Cumulative_Amount	Year_Rank
1	2	2024	34075.16	55103.30	1
2	2	2023	16178.14	21028.14	2
3	2	2022	4850.00	4850.00	3
4	3	2024	92153.88	92153.88	1
5	4	2023	666804.00	666804.00	1
6	5	2024	5760.00	33613.96	2
7	5	2023	27853.96	27853.96	1
8	6	2024	54439.00	54439.00	1
9	7	2024	370399.32	384205.79	1
10	7	2023	13806.47	13806.47	2
11	8	2024	5604.29	10446.69	1
12	8	2023	4842.40	4842.40	2
13	9	2024	11462.40	45710.18	2
14	9	2023	34247.78	34247.78	1
15	10	2024	1333.84	10224.43	2
16	10	2022	8890.59	8890.59	1
17	12	2024	18686.45	25285.30	1
18	12	2023	6598.85	6598.85	2
19	13	2025	268941.94	363262.43	1
20	13	2024	65125.49	94320.49	2
21	13	2023	29195.00	29195.00	3
22	14	2024	17388.00	34536.44	1
23	14	2023	17148.44	17148.44	2
24	16	2024	193793.64	193793.64	1
25	19	2024	10670.00	11060.25	1
26	19	2023	390.25	390.25	2
27	20	2024	3099.30	10635.54	2
28	20	2023	7536.24	7536.24	1

פרויקט בסיסי נתונים חלק ב'	אוניברסיטת בן גוריון בנגב	
	המחלקה להנדסת תעשייה וניהול	

2. שאילתה המשלבת פונקציית חלון שימוש ב **LAG()** ו **ROW NUMBER()**:
עבור השכרות של סוג רכב מסוים בסניף מסוים מה המספר הסידורי של הרכב לפי המחיר הגבוהה לנמוך, בנוסף עבור כל סוג רכב בסניף הצג מה היה מחירו בהשכרה הקודמת.

• הגיון עסקי:

המטרה העסקית של השאילתה היא לעקוב אחרי שינויים במחיר השכרת רכבים מאותו הסוג בסניף בו מבוצעת ההשכרה, ולהשוות את המחיר הנוכחי של רכב למחיר שהיה בהשכרה הקודמת. כך ניתן לזהות מגמות במחירים (האם המחיר עלה או ירד), ולהבין אם יש צורך לבצע התאמות במחירים על פי נתוני השוק. בנוסף השאילתה מדרגת את ההשכרה לפי מכיר השכרת הרכב ליום בסדר עולה מהמחיר הגבוה לנמוך.

SELECT

B.BranchID, R.PricePerDay, C.CarType,
S.startdate,

pricegroup=**ROW_NUMBER()** OVER (**PARTITION BY** B.BranchID **ORDER BY** R.PricePerDay desc),
prev_price=**LAG**(R.PricePerDay) OVER (**PARTITION BY** B.BranchID, C.CarType **ORDER BY** S.startdate asc)

FROM

RESULTS AS R JOIN CARS AS C ON R.CarID = C.CarID JOIN
SEARCHES AS S ON R.SearchIP = S.SearchIP AND R.SearchDT = S.SearchDT
JOIN BRANCHES AS B ON C.BranchID = B.BranchID

ORDER BY

B.BranchID, pricegroup;

• פלט השאילתה: 2125 רשומות



	BranchID	PricePerDay	CarType	startdate	pricegroup	prev_price
1	2	847.00	BMW 3 Series	2023-02-10	1	485.00
2	2	485.00	BMW 3 Series	2023-02-08	2	NULL
3	2	485.00	BMW 3 Series	2023-09-20	3	847.00
4	3	5115.00	Subaru Forester	2023-02-13	1	NULL
5	4	55567.00	Honda CR-V	2023-02-08	1	NULL
6	5	520.00	Mazda CX-5	2023-01-30	1	NULL
7	5	480.00	Kia Sorento	2023-02-04	2	NULL
8	5	480.00	Kia Sorento	2024-11-13	3	480.00
9	6	7777.00	Audi A4	2023-02-02	1	NULL
10	7	445.00	BMW X5	2023-02-16	1	NULL
11	7	253.00	Toyota Corolla	2023-02-03	2	236.00
12	7	253.00	Toyota Corolla	2024-07-19	3	253.00
13	7	236.00	Toyota Corolla	2020-06-24	4	NULL
14	8	161.00	Mercedes GLE	2024-05-06	1	NULL
15	8	161.00	Mercedes GLE	2024-11-13	2	161.00
16	9	983.00	BMW X3	2023-01-30	1	NULL
17	9	731.00	Ford Explorer	2023-02-16	2	254.00
18	9	477.00	Subaru Impreza	2023-01-18	3	NULL
19	9	254.00	Ford Explorer	2023-01-30	4	NULL
20	9	254.00	Ford Explorer	2023-11-28	5	731.00
21	10	493.00	Toyota Corolla	2023-01-25	1	NULL
22	10	184.00	Honda Accord	2024-03-18	2	NULL
23	12	5543.00	Honda CR-V	2021-05-19	1	NULL
24	12	472.00	Audi A4	2023-01-30	2	NULL
25	12	398.00	Honda CR-V	2023-02-10	3	5543.00

פרויקט בסיסי נתונים חלק ב'	אוניברסיטת בן גוריון בנגב המחלקה להנדסת תעשייה וניהול
-------------------------------	--



שאלתה מקוננת תוך שימוש ב-CTE (פסקת WITH) (10%)

השאלתה תתמקד בחישוב פרטים הקשורים לחיפוש, מכירות והפסדים אפשריים.

1. מספר החיפושים שלא הובילו להזמנה.
 2. חישוב משך השכרת רכב ממוצע עבור חיפושים שהובילו להזמנה.
 3. חישוב ממוצע ההכנסות עבור יום השכרה, על סמך ההזמנות שבוצעו.
 4. הפקת דו"ח הכולל הערכת הפסדים פוטנציאליים עבור חיפושים שלא הובילו להזמנה.
- **הגיון עסקי:** חיפושים שלא הובילו להזמנה חושפים חולשות בתהליך המכירה, משך ההשכרה הממוצע מגלה מגמות שימוש ממוצעות של לקוחות אווים, חישוב ההפסדים הפוטנציאליים הוא כלי חשוב שיכול לתרום להנעת העובדים לקידום מכירות החברה ויכול לשמש חזון להצבת יעדים לחברה בהמשך.

With

dismissedSearches as (חישוב מספר החיפושים לא הובילו להזמנה---

select

dismissed_num=count (*)

from SEARCHES as S left join RESULTS as R on S.SearchDT=R.SearchDT and
S.searchIP=R.SearchIP

left join EXTRAS as E on E.SearchIP=R.SearchIP and E.SearchDT=R.SearchDT and
E.ResultID=R.ResultID

left join EXTRASOFRESERVATION as EX on E.SearchIP=EX.SearchIP and
E.SearchDT=EX.SearchDT and E.ResultID=EX.ResultID and E.Extra=EX.Extra

left join RESERVATIONS as RE on RE.ReservationID=EX.ReservationID

where RE.ReservationID is null

),

days_rent as (מספר ימי ההזמנות הממוצע להשכרת רכב---

select

Reservation_Num=count (*),

Avg_Rent_Dayes=avg(datediff(dd,S.startDate,S.FinishDate))

from SEARCHES as S join RESULTS as R on S.SearchDT=R.SearchDT and
S.searchIP=R.SearchIP

join EXTRAS as E on E.SearchIP=R.SearchIP and E.SearchDT=R.SearchDT and
E.ResultID=R.ResultID

join EXTRASOFRESERVATION as EX on E.SearchIP=EX.SearchIP and
E.SearchDT=EX.SearchDT and E.ResultID=EX.ResultID and E.Extra=EX.Extra

join RESERVATIONS as RE on RE.ReservationID=EX.ReservationID

),


Avg_Dismissed_Income as (חישוב ממוצע ההכנסות עבור יום השכרה---

select

Avg_Income_Of_Rental_Day=round(AVG(R.pricePerDay),2)

from SEARCHES as S join RESULTS as R on S.SearchDT=R.SearchDT and
S.searchIP=R.SearchIP

join EXTRAS as E on E.SearchIP=R.SearchIP and E.SearchDT=R.SearchDT and
E.ResultID=R.ResultID

פרויקט בסיסי נתונים חלק ב'	אוניברסיטת בן גוריון בנגב	
	המחלקה להנדסת תעשייה וניהול	

```

join EXTRASOFRESERVATION as EX on E.SearchIP=EX.SearchIP and
E.SearchDT=EX.SearchDT and E.ResultID=EX.ResultID and E.Extra=EX.Extra
join RESERVATIONS as RE on RE.ReservationID=EX.ReservationID
)
select DS.dismissed_num,DR.Avg_Rent_Dayes,AI.Avg_Income_Of_Rental_Day,
Avg_Dismissed_Income=round(DS.dismissed_num*DR.Avg_Rent_Dayes*AI.Avg_Income_Of_
Rental_Day,2)
from dismissedSearches as DS join days_rent as DR on 1=1
join Avg_Dismissed_Income as AI on 1=1;

```

● פלט השאילתה:

	dismissed_num	Avg_Rent_Dayes	Avg_Income_Of_Rental_Day	Avg_Dismissed_Income
1	1026	135	29108.310000	4031792018.100000

פרויקט בסיסי נתונים חלק ב'	אוניברסיטת בן גוריון בנגב המחלקה להנדסת תעשייה וניהול
-------------------------------	--



מטלה 2 (35%) – יישומי כלים מתקדמים

View (5%)

VIEW עבור סיכום ההכנסות השנתיות לסניפים: נניח שברצון המנכ"ל לחשב את ההכנסות השנתיות לכל סניף, כולל ההזמנות שבוצעו והתוספות שהוזמנו. זה יכול לעזור למנהלי סניפים להבין את ביצועיהם ביחס לשנים.

מורכבות השאילתה:

ה-VIEW מחלץ נתונים ממספר טבלאות (RESERVATIONS, EXTRASOFRESERVATION, RESULTS) ומשתמש בפונקציות כמו SUM, DATEDIFF (BRANCHES, SEARCHES). בכך הוא עונה על הקריטריון של שאילתה מורכבת שמבוצעת לעיתים קרובות.

```
CREATE VIEW vw_Yearly_Revenue_Per_Branch AS
SELECT
  B.BranchID,
  B.Location,
  Year_Of_Reservation= YEAR(R.ReservationDate) ,
  Total_Revenue=SUM(RE.PricePerDay * DATEDIFF(DAY, S.StartDate, S.FinishDate) + EXR.Qty
  * E.Price)
FROM
  reservations as R left join extrasofreservation
  as EXR on EXR.reservationID=R.reservationID
  join RESULTS AS RE ON RE.ResultID=EXR.ResultID
  join cars as c on c.CarID=re.CarID
  join searches as S on S.SearchIP=RE.SearchIP
  left join EXTRAS as E on E.SearchIP=S.SearchIP
  join BRANCHES as B on B.BranchID=C.BranchID
GROUP BY
  B.BranchID, B.Location, YEAR(R.ReservationDate);
```

פרויקט בסיסי נתונים חלק ב'	אוניברסיטת בן גוריון בנגב	
	המחלקה להנדסת תעשייה וניהול	

פונקציות (Functions) (10%, 2 פונקציות – 5% לכל פונקציה)

Tabular Function - פונקציה המחזירה טבלה:

הפונקציה תחשב את ממוצע ההכנסות השנתי לכל סניף לפי שנה מסוימת, ותדרג את ממוצע המחירות של הסניף לאורך השנים על בסיס הממוצע.

הגיון עסקי:

ההגיון העסקי מאחורי הפונקציה הוא שיפור ניהול והבנת ביצועי הסניפים בעסק. הפונקציה מאפשרת להעריך ולהשוות את הביצועים של סניפים שונים לפי הכנסותיהם השנתיות, ובכך להקל על קבלת החלטות עסקיות.

```
create function Get_Branch_Yearly_Sales_rank_VW (@BranchID int)
returns table
as
return
(select BranchID,
avg_yearly_sales=avg(Total_Revenue),
from vw_Yearly_Revenue_Per_Branch
where (BranchID=@BranchID)
group by BranchID
)
```

דוגמא לשימוש בפונקציה:

```
SELECT *
FROM dbo.Get_Branch_Yearly_Sales_Rank_VW (9)
```

פלט השאילתה:

	BranchID	avg_yearly_sales
1	9	25816.990000

פרויקט בסיסי נתונים חלק ב'	אוניברסיטת בן גוריון בנגב המחלקה להנדסת תעשייה וניהול
-------------------------------	--



Scalar Function - פונקציה המחזירה סקלר:

הפונקציה CarAvailability נועדה לשפר את ניהול צי הרכבים של החברה ולספק מידע מדויק על הזמינות של רכבים בסניפים בזמן אמת.

הגיון עסקי:

מתן מידע בזמן אמת על מספר הרכבים הזמינים. זה מאפשר למנהלי סניפים לנהל את הצי בצורה אופטימלית, למנוע טעויות בהזמנות, ולהבטיח שיפור בשירות הלקוחות.

```
CREATE FUNCTION CarAvailability (@BRANCHID INT, @CHECKDATE DATE)
RETURNS int
AS BEGIN
    DECLARE @BRANCHCARS Int
    SELECT @BRANCHCARS = COUNT (CARS.BranchID)
    FROM CARS
    WHERE CARS.BranchID = @BRANCHID

    DECLARE @RentedCars INT;
    SELECT @RentedCars = COUNT(*)
    FROM SEARCHES AS S JOIN RESULTS AS R ON
    R.SearchIP=S.SearchIP
    JOIN EXTRAS AS E ON
    E.ResultID=R.ResultID
    JOIN CARS AS C ON
    C.CarID=R.CarID
    JOIN EXTRASOFRESERVATION AS ER ON
    ER.ResultID=E.ResultID
    JOIN RESERVATIONS AS RES ON
    ER.ReservationID=RES.ReservationID
    WHERE C.BranchID = @BranchID
    AND @CHECKDATE BETWEEN S.StartDate AND S.FinishDate

    SET @BRANCHCARS=@BRANCHCARS-@RentedCars ;
    RETURN @BRANCHCARS
END
```

דוגמה לשימוש בפונקציה:

```
SELECT
    AvailabilityCARS= dbo.CarAvailability (BranchID, GETDATE())
FROM BRANCHES
WHERE
    BranchID = 418
```

פלט השאילתה:

פרויקט בסיסי נתונים חלק ב'	אוניברסיטת בן גוריון בנגב המחלקה להנדסת תעשייה וניהול
-------------------------------	--



AvailabilityCARS

3

Trigger (10%)

הטריגר מופעל בעקבות הוספה של מכונית לסניף מסוים או הורדת מכונית בשל סיבה מסוימת. המטרה הינה לדעת כמה מכוניות שייכות לכל סניף. הסיבה שאנו רוצים לדעת כמה מכוניות יש בכל סניף היא בשביל לדעת אם יש מחסור במכוניות בסניף מסוים או להפך יש יותר מדי מכוניות ואין ביקוש למכוניות וניתן להעביר מכוניות לסניף אחר. בשביל להמחיש את הטריגר הוספנו שדה מחושב בטבלת סניפים אשר סופר כמה מכוניות יש בכל סניף.

```
ALTER TABLE BRANCHES
ADD AvailableCars INT NOT NULL DEFAULT 0;
UPDATE BRANCHES
SET AvailableCars = (
    SELECT COUNT(*)
    FROM CARS
    WHERE CARS.BranchID = BRANCHES.BranchID
)
```

```
--DROP TRIGGER IF EXISTS AfterCarInsert
CREATE TRIGGER AfterCarInsert
ON CARS for INSERT AS UPDATE BRANCHES
SET AvailableCars = ( ISNULL(AvailableCars,0)+(
    SELECT COUNT(*)
    FROM INSERTED as I
    WHERE I.BRANCHID = BRANCHES.BranchID)
)
```

```
--DROP TRIGGER IF EXISTS AfterCarDelete
CREATE TRIGGER AfterCarDelete
ON CARS for DELETE AS UPDATE BRANCHES
SET AvailableCars = ( ISNULL(AvailableCars,0)-(
    SELECT COUNT(*)
    FROM deleted as D
    WHERE D.BRANCHID = BRANCHES.BranchID))
```

מימוש הטריגר:

```
INSERT INTO CARS (CarID, Gear, Color, BranchID, CarType)
```

פרויקט בסיסי נתונים חלק ב'	אוניברסיטת בן גוריון בנגב	
	המחלקה להנדסת תעשייה וניהול	

```
VALUES ('444', 'Automatic', 'Red', 2, 'Kia Sorento'),
('781', 'Automatic', 'Red', 2, 'Kia Sorento'),
('54189', 'Automatic', 'Red', 2, 'Kia Sorento'),
('88997', 'Automatic', 'Red', 2, 'Kia Sorento');
DELETE FROM CARS WHERE CarID = '781'
DELETE FROM CARS WHERE CarID = '54189'
```

```
DELETE FROM CARS WHERE CarID = '444';
SELECT * FROM BRANCHES WHERE BranchID = 2
```

לפני הטריגר:

	BranchID	AvailableCars
1	2	6

אחרי הטריגר:

	BranchID	AvailableCars
1	2	8

(994 rows affected)

(4 rows affected)

(994 rows affected)

(1 row affected)

(994 rows affected)

(1 row affected)

Completion time: 2025-01-25T22:28:15.3525514+02:00

פרויקט בסיסי נתונים חלק ב'	אוניברסיטת בן גוריון בנגב	
	המחלקה להנדסת תעשייה וניהול	

פרוצדורה שמורה (Stored Procedure) (10%)

הגיון עסקי:

פרוצדורה זו עוזרת לזהות את הלקוחות שמניבים את הרווח הגבוה ביותר לעסק. זיהוי זה מאפשר למקד מאמצי שיווק ושימור לקוחות בלקוחות המובילים. בנוסף, בזכות מידע זה ניתן לספק תובנות לבניית תוכניות נאמנות מותאמות אישית כגון הנחות או מבצעים ללקוחות שמוציאים סכומים גבוהים וכך מחזקת את הקשר עם לקוחות אלו ומונעת מעבר שלהם למתחרים.

```

DROP PROCEDURE GetTopCustomers
CREATE PROCEDURE GetTopCustomers
    @TopCount INT
AS
BEGIN
    SELECT TOP (@TopCount)
        R.Email,
        TotalSpent=SUM(RT.PricePerDay * ER.Qty + E.Price * ER.Qty)
    FROM RESERVATIONS AS R
    JOIN EXTRASOFRESERVATION ER ON R.ReservationID = ER.ReservationID
    JOIN RESULTS AS RT ON ER.ResultID = RT.ResultID
    JOIN EXTRAS AS E ON ER.ResultID = E.ResultID
    GROUP BY R.Email
    ORDER BY SUM(RT.PricePerDay * ER.Qty + E.Price * ER.Qty) DESC;
END;

```

מימוש הפרוצדורה:

```
EXEC GetTopCustomers @TopCount = 5;
```

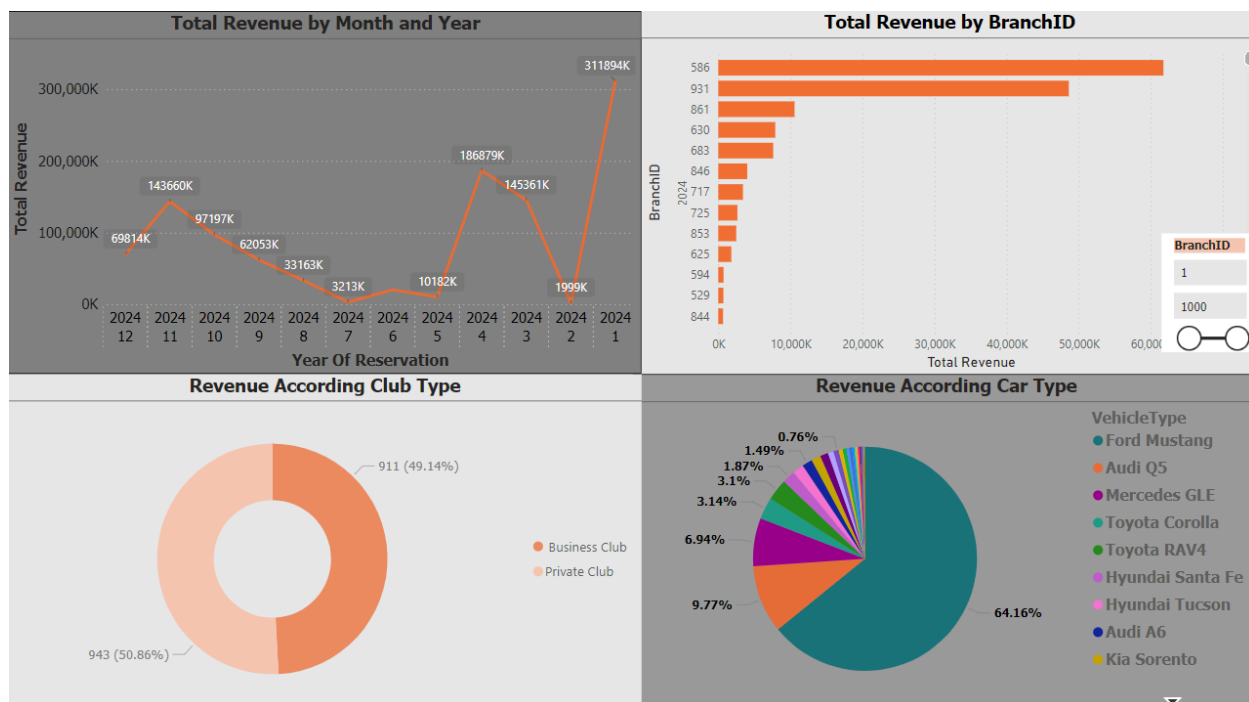
פלט השאילתה:

	Email	TotalSpent
1	jopy8a@flickr.com	35917927.62
2	btozeraa@umich.edu	26012982.60
3	lberkey5z@hud.gov	15607873.71
4	lshadrachlt@marketwatch.com	11972642.54
5	ajermin9x@shop-pro.jp	8000065.08



מטלה 3 (20%) – כלים להצגת נתונים

דוח עסקי לסמנכ"ל התפעול סמנכ"ל הכספים, מנהלי שיווק וצוותי תפעול:



• בקרה על הכנסות:

הצגת סך ההכנסות לפי חודשים ("Total Revenues 2024") מאפשרת לעקוב אחר ביצועי החברה לאורך השנה ולזהות עונתיות או מגמות בירידה/עלייה בהכנסות.

• ניתוח סניפים:

גרף "Total Revenue by Branches" מציג את ההכנסות הגבוהות ביותר לפי סניף, ומספק תובנות על סניפים מצליחים או כאלה שדורשים תשומת לב לשיפור.

• העדפות לקוחות לפי סוג רכב:

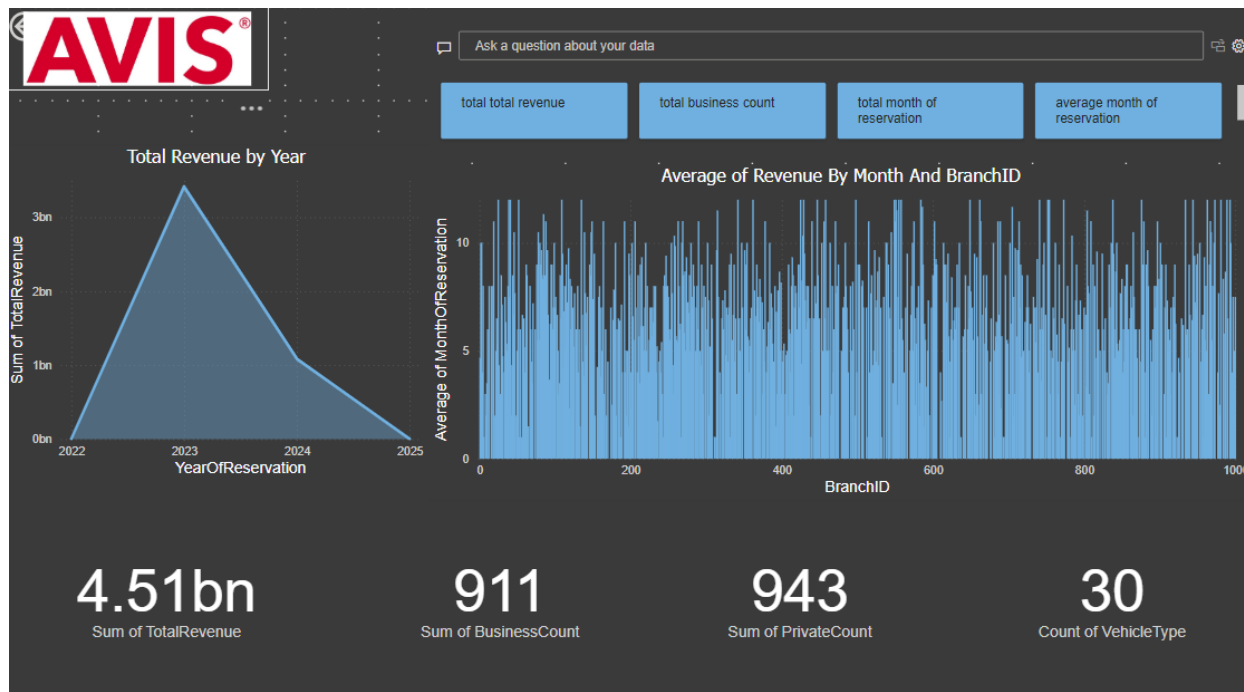
הפאי-צ'ארט "Revenue according Car Type" מאפשר להבין אילו סוגי רכבים מניבים את ההכנסות הגבוהות ביותר, וכך לספק המלצות על מיקוד במלאי או במבצעי שיווק.

• פילוח לקוחות לפי מועדון:

גרף "Total Reservations by Customer Club Type" עוזר להבחין בין לקוחות עסקיים לפרטיים, כדי לזהות את משקל כל קבוצה ולטייב שירותים והצעות ייעודיות.



לוח מחוונים למנכ"ל:



- **גרף סך ההכנסות לפי שנים: (Total Revenue by Year)**
הגרף בצד שמאל למעלה מציג את סך ההכנסות לפי שנת ההזמנה. ניתן לראות מגמה של גידול משמעותי בהכנסות בשנת 2023 ולאחר מכן ירידה חדה ב-2024 ו-2025.
- **גרף ממוצע הכנסות לפי חודש וסניף:**
הגרף בצד ימין למעלה מציג את ממוצע ההכנסות החודשי לפי מזהה הסניף (BranchID). ככל הנראה, הוא משמש להשוואה בין סניפים שונים על בסיס הכנסות חודשיות.
- **כרטיסי מידע בתחתית:**

4.51bn (סך ההכנסות): מציג את סך ההכנסות הכולל.

911 (סך ההזמנות העסקיות): מציג את כמות ההזמנות שבוצעו על ידי לקוחות עסקיים.

943 (סך ההזמנות הפרטיות): מציג את כמות ההזמנות שבוצעו על ידי לקוחות פרטיים.

30 (סוגי רכבים): מציג את מספר סוגי הרכבים הקיימים בהיצע של אווים.



קוד עזר להפקת הדוחות:

```

REATE OR ALTER VIEW ReservationRevenue AS
SELECT
    B.BranchID,
    B.Location AS BranchLocation,
    YearOfReservation = YEAR(RES.ReservationDate),
    MonthOfReservation = MONTH(RES.ReservationDate),
    VehicleType = CT.CarType,
    TotalRevenue = SUM(RE.pricePerDay * DATEDIFF(DAY, S.StartDate, S.FinishDate)) +
    SUM(EXR.Qty * E.Price),
    TotalReservations = COUNT(DISTINCT RES.ReservationID),
    BusinessCount = COUNT(CASE WHEN S.Club = 1 THEN 1 END),
    PrivateCount = COUNT(CASE WHEN S.Club = 0 THEN 1 END),
    CustomerType = (CASE WHEN S.Club = 1 THEN 'Business' WHEN S.Club = 0 THEN
'Private'ELSE 'Unknown'END)
FROM
    SEARCHES S
JOIN
    RESULTS RE
    ON S.SearchIP = RE.SearchIP AND S.SearchDT = RE.SearchDT
JOIN
    BRANCHES B
    ON S.BranchT = B.BranchID
JOIN
    CARS C
    ON RE.CarID = C.CarID
JOIN
    CARTYPES CT
    ON C.CarType = CT.CarType
JOIN
    EXTRASOFRESERVATION EXR
    ON RE.ResultID = EXR.ResultID
    AND RE.SearchIP = EXR.SearchIP
    AND RE.SearchDT = EXR.SearchDT
JOIN
    EXTRAS E
    ON EXR.Extra = E.Extra
    AND EXR.ResultID = E.ResultID
    AND EXR.SearchIP = E.SearchIP
    AND EXR.SearchDT = E.SearchDT
JOIN
    RESERVATIONS RES
    ON RES.ReservationID = EXR.ReservationID

```

פרויקט בסיסי נתונים חלק ב'	אוניברסיטת בן גוריון בנגב	
	המחלקה להנדסת תעשייה וניהול	

GROUP BY

B.BranchID,
 B.Location,
 CT.CarType,
 YEAR(RES.ReservationDate),
 S.Club,
 MONTH(RES.ReservationDate);

פרויקט בסיסי נתונים חלק ב'	אוניברסיטת בן גוריון בנגב המחלקה להנדסת תעשייה וניהול
-------------------------------	--



מטלה 4 (10%) – אופטימיזציה של שאלות באמצעות Generative AI

- השאלתה הראשונה שבחרנו לעשות לה אופטימיזציה הינה השאלתה הבאה :

```
select c.branchID,
year_avg_income=avg(RE.pricePerDay*
DATEDIFF(DAY,S.StartDate,S.FinishDate)+EXR.Qty*E.Price)
from reservations as R left join extrasofreservation
as EXR on EXR.reservationID=R.reservationID
join RESULTS AS RE ON RE.ResultID=EXR.ResultID
join cars as c on c.CarID=re.CarID
join searches as S on S.SearchIP=RE.SearchIP AND S.SearchDT=RE.SearchDT
join EXTRAS as E on E.SearchIP=S.SearchIP AND S.SearchDT=E.SearchDT AND E.ResultID =
RE.ResultID
group by C.BranchID
having avg(RE.pricePerDay* DATEDIFF(DAY,S.StartDate,S.FinishDate)+EXR.Qty*E.Price)<
(
select avg(RE.pricePerDay* DATEDIFF(DAY,S.StartDate,S.FinishDate)+EXR.Qty*E.Price)
from reservations as R join extrasofreservation
as EXR on EXR.reservationID=R.reservationID
join RESULTS AS RE ON RE.ResultID=EXR.ResultID
join cars as c on c.CarID=re.CarID
join searches as S on S.SearchIP=RE.SearchIP AND S.SearchDT=RE.SearchDT
join EXTRAS as E on E.SearchIP=S.SearchIP and S.SearchDT=E.SearchDT AND E.ResultID =
RE.ResultID
)
order by year_avg_income desc
```

השאלתה אשר הבינה המלאכותית בנתה:

```
CREATE INDEX idx_reservations_reservationid ON RESERVATIONS(ReservationID);
CREATE INDEX idx_extrasofreservation ON EXTRASOFRESERVATION(ReservationID, ResultID,
Qty);
CREATE INDEX idx_results ON RESULTS(ResultID, SearchIP, SearchDT, pricePerDay, CarID);
CREATE INDEX idx_cars ON CARS(CarID, BranchID);
CREATE INDEX idx_searches ON SEARCHES(SearchIP, SearchDT);
CREATE INDEX idx_extras ON EXTRAS(SearchIP, SearchDT, ResultID, Price);
```



```

SELECT
    C.BranchID,
    AVG(RE.pricePerDay * DATEDIFF(DAY, S.StartDate, S.FinishDate) + EXR.Qty * E.Price) AS
year_avg_income
FROM
    RESERVATIONS AS R
    JOIN EXTRASOFRESERVATION AS EXR ON EXR.ReservationID = R.ReservationID
    JOIN RESULTS AS RE ON RE.ResultID = EXR.ResultID
    JOIN CARS AS C ON C.CarID = RE.CarID
    JOIN SEARCHES AS S ON S.SearchIP = RE.SearchIP AND S.SearchDT = RE.SearchDT
    JOIN EXTRAS AS E ON E.SearchIP = S.SearchIP AND E.SearchDT = S.SearchDT AND
E.ResultID = RE.ResultID
GROUP BY
    C.BranchID
HAVING
    AVG(RE.pricePerDay * DATEDIFF(DAY, S.StartDate, S.FinishDate) + EXR.Qty * E.Price) <
(
    SELECT
        AVG(RE.pricePerDay * DATEDIFF(DAY, S.StartDate, S.FinishDate) + EXR.Qty * E.Price)
    FROM
        RESERVATIONS AS R
        JOIN EXTRASOFRESERVATION AS EXR ON EXR.ReservationID = R.ReservationID
        JOIN RESULTS AS RE ON RE.ResultID = EXR.ResultID
        JOIN CARS AS C ON C.CarID = RE.CarID
        JOIN SEARCHES AS S ON S.SearchIP = RE.SearchIP AND S.SearchDT = RE.SearchDT
        JOIN EXTRAS AS E ON E.SearchIP = S.SearchIP AND E.SearchDT = S.SearchDT AND
E.ResultID = RE.ResultID
)
ORDER BY
    year_avg_income DESC;

```

פרויקט בסיסי נתונים חלק ב'	אוניברסיטת בן גוריון בנגב	
	המחלקה להנדסת תעשייה וניהול	

סיכום האופטימיזציות שבוצעו ע"י הכלי:

1. מניעת חישובים כפולים:

בשאלתה הראשונית החישוב של הממוצע (AVG) נעשה פעמיים: גם ב-HAVING וגם בתת-השאלתה.

2. שימוש ב-CTE:

פיצול השאלתה למקטעים לוגיים:

- IncomeData מחשב את הממוצע השנתי (year_avg_income) עבור כל סניף.
- OverallAverageIncome מחשב את הממוצע הכללי (overall_avg_income) על בסיס הנתונים מ-IncomeData.
- מאפשר קריאות טובה יותר וניהול קל של השאלתה.

3. אינדקסים מותאמים:

המטרה היא לשפר ביצועים של פעולות JOIN, GROUP BY ושליפת נתונים.

קישור לשיחה עם ה-CHATGPT :

<https://chatgpt.com/share/67952df0-66b8-800b-bb5b-f6b6f82d749c>

פלט השאלתה:

	BranchID	year_avg_income
1	435	1421972.170000
2	833	1305235.211428
3	129	1019198.170000
4	778	925655.820000
5	883	906373.846428
6	12	868489.771428
7	570	717536.240000
8	127	682078.800000
9	73	681912.000000
10	896	681456.600000
11	132	675787.588750
12	4	666804.000000
13	921	652700.075000
14	535	594048.000000
15	538	593642.680000
16	847	568966.800000
17	827	565076.785000
18	529	538698.516000
19	323	536975.370000

פרויקט בסיסי נתונים חלק ב'	אוניברסיטת בן גוריון בנגב המחלקה להנדסת תעשייה וניהול
-------------------------------	--



- השאילתה השנייה שבחרנו לייעל הינה השאילתה הבאה:

```
SELECT S.SearchDT,S.SearchIP,S.StartDate,S.FinishDate,S.BranchT,S.BranchR,R.ResultID,
E.Extra
FROM SEARCHES AS S LEFT JOIN RESULTS AS R ON S.SearchIP = R.SearchIP
AND S.SearchDT = R.SearchDT LEFT JOIN
EXTRAS AS E ON R.SearchIP = E.SearchIP
AND R.SearchDT = E.SearchDT
AND R.ResultID = E.ResultID LEFT JOIN
EXTRASOFRESERVATION AS ER ON E.SearchIP = ER.SearchIP
AND E.SearchDT = ER.SearchDT
AND E.ResultID = ER.ResultID
AND E.Extra = ER.Extra
WHERE ER.ReservationID IS NULL
ORDER BY S.SearchDT DESC
```

השאילתה לאחר אופטימיזציה של הבינה המלאכותית:

-- יצירת אינדקסים

```
CREATE INDEX idx_searches_search ON SEARCHES(SearchIP, SearchDT, StartDate, FinishDate,
BranchT, BranchR);
CREATE INDEX idx_results_search ON RESULTS(SearchIP, SearchDT, ResultID);
CREATE INDEX idx_extras_search ON EXTRAS(SearchIP, SearchDT, ResultID, Extra);
CREATE INDEX idx_extrasofreservation_search ON EXTRASOFRESERVATION(SearchIP,
SearchDT, ResultID, Extra, ReservationID);
```

-- שאילתה אופטימלית

```
SELECT
    S.SearchDT,
    S.SearchIP,
    S.StartDate,
    S.FinishDate,
    S.BranchT,
    S.BranchR,
    R.ResultID,
    E.Extra
FROM
    SEARCHES AS S
LEFT JOIN
    RESULTS AS R
ON
```



```

S.SearchIP = R.SearchIP
AND S.SearchDT = R.SearchDT
LEFT JOIN
  EXTRAS AS E
ON
  R.SearchIP = E.SearchIP
  AND R.SearchDT = E.SearchDT
  AND R.ResultID = E.ResultID
WHERE
  NOT EXISTS (
    SELECT 1
    FROM EXTRASOFRESERVATION AS ER
    WHERE
      E.SearchIP = ER.SearchIP
      AND E.SearchDT = ER.SearchDT
      AND E.ResultID = ER.ResultID
      AND E.Extra = ER.Extra
  )
ORDER BY
  S.SearchDT DESC;

```

קישור לשיחה עם הבינה המלאכותית:

<https://chatgpt.com/share/679380df-e298-800f-9b59-55b1c87b9efb>

הצעות לאופטימיזציה:

1. שימוש ב-EXISTS במקום JOIN:

לעיתים קרובות, שימוש ב-EXISTS במקום צירופים מיותרים יכול לשפר את הביצועים. כך נוכל למנוע איחוד טבלאות שאיננו הכרחי ולהתמקד בסינון הרשומות. עם LEFT JOIN: השאילתה מבצעת צירוף של שתי הטבלאות ואז מסננת את התוצאות. כל הנתונים מהצירוף נשמרים בזיכרון, גם אם הם לא נחוצים. עם NOT EXISTS: השאילתה רק בודקת אם קיימת התאמה, והיא מפסיקה ברגע שמוצאת רשומה מתאימה, ללא צירוף מלא.

2. הוספת אינדקסים:

ודא שהעמודות שמשמשות בתנאים של הצירופים והסינון (כמו SearchIP, SearchDT, ResultID, Extra), מכוסות על ידי אינדקסים.

3. השימוש ב-SELECT 1:

עשוי להיות טיפה יותר מהיר מכיוון שאינו צריך לקרוא או לעבד נתונים ספציפיים מהטבלה. עם זאת, ברוב מערכות ה-SQL ההבדל בביצועים הוא זניח. בעצם SELECT 1 הוא יותר "מפורש" ומרמז למנוע ה-SQL שאתה לא באמת צריך מידע מתוך הטבלה הפנימית, אלא רק את קיום הרשומה.



פלט השאילתה :

	SearchDT	SearchIP	StartDate	FinishDate	BranchT	BranchR	ResultID	Extra
1	2025-01-16 19:50:00.0000000	68.206.201.85	2025-02-09	2025-02-21	163	140	NULL	NULL
2	2025-01-14 11:00:00.0000000	130.167.81.81	2025-02-09	2025-03-14	234	333	6854192	No Extra
3	2025-01-13 08:43:00.0000000	135.105.53.182	2025-02-05	2025-03-22	878	386	4897561	First Aid Kit
4	2025-01-12 11:38:00.0000000	13.212.35.36	2025-02-09	2025-03-01	353	18	3151262	Baby Seat
5	2025-01-10 21:08:00.0000000	89.223.252.105	2025-01-23	2025-03-01	402	561	NULL	NULL
6	2025-01-09 03:58:00.0000000	13.83.224.31	2025-01-16	2025-03-02	909	957	4776798	WiFi
7	2025-01-08 20:55:00.0000000	207.165.237.107	2025-01-13	2025-01-21	698	75	NULL	NULL
8	2025-01-07 14:06:00.0000000	208.31.127.222	2025-01-08	2025-02-08	951	516	NULL	NULL
9	2025-01-07 05:12:00.0000000	52.86.153.166	2025-01-18	2025-02-04	801	206	NULL	NULL
10	2025-01-06 05:30:00.0000000	213.137.238.123	2025-02-04	2025-03-16	109	97	NULL	NULL
11	2025-01-02 07:15:00.0000000	56.181.89.139	2025-01-08	2025-01-29	135	923	NULL	NULL
12	2025-01-02 04:49:00.0000000	13.193.43.171	2025-01-19	2025-02-11	764	522	7328650	WiFi
13	2025-01-01 13:56:00.0000000	39.162.219.43	2025-01-17	2025-02-08	249	484	NULL	NULL
14	2024-12-29 02:42:00.0000000	218.83.163.181	2025-01-28	2025-02-28	858	823	NULL	NULL
15	2024-12-28 08:53:00.0000000	123.238.9.226	2025-01-11	2025-02-14	369	282	3316007	No Extra
16	2024-12-25 23:43:00.0000000	128.28.57.132	2025-01-04	2025-01-31	60	898	7051048	WiFi
17	2024-12-22 13:32:00.0000000	90.33.254.16	2025-01-02	2025-01-30	844	121	NULL	NULL
18	2024-12-19 14:49:00.0000000	49.151.127.146	2025-01-06	2025-02-13	185	898	NULL	NULL
19	2024-12-19 11:35:00.0000000	21.190.89.134	2024-12-30	2025-01-28	864	140	NULL	NULL
20	2024-12-16 15:09:00.0000000	53.187.89.164	2024-12-27	2025-02-04	456	385	NULL	NULL
21	2024-12-11 10:37:00.0000000	73.45.219.184	2024-12-19	2024-12-26	70	637	5960804	Baby Seat
22	2024-12-07 05:16:00.0000000	70.224.173.103	2024-12-23	2025-01-12	42	943	8192539	Baby Seat

פרויקט בסיסי נתונים חלק ב'	אוניברסיטת בן גוריון בנגב המחלקה להנדסת תעשייה וניהול
-------------------------------	--



פרק שני - בונס

מה שהטכניקה עושה זה בעצם המרת שורות (בקוד שלנו נתונים של כל חודש) לעמודות. זה יכול להיות מאוד שימושי אם אנחנו רוצים לראות את הנתונים בצורה שמקלה על הקריאה וההשוואה בין חודשים שונים, במקום שיהיה עלינו לחפש את כל המידע בתוך שורות נפרדות. הקוד משתמש ב-VIEW שייצרנו במטלה 3 להצגת הנתונים דרך ה-PowerBI. ומציג לנו את כלל הסניפים ואת ההכנסה שלהם עבור כל חודש ב-2024 בתצוגה שנעימה וברורה לעין. איך זה עובד?

1. **נתונים מקוריים:** יש לנו טבלה (כמו ReservationRevenue) בה כל שורה מייצגת הכנסה של סניף בחודש מסוים.
2. **הפיכת שורות לעמודות:** באמצעות PIVOT, אנו לוקחים את הנתונים מהשדות MonthOfReservation ו-TotalRevenue וממירים אותם לעמודות, כך שכל חודש מקבל עמודה משלו. השיטה הזו מאפשרת הצגה נוחה יותר של נתונים חודשיים.

הקוד למימוש:

SELECT

```
BranchID,
January=COALESCE([1], 0),
February=COALESCE([2], 0),
March=COALESCE([3], 0),
April=COALESCE([4], 0),
May=COALESCE([5], 0),
June=COALESCE([6], 0),
July=COALESCE([7], 0),
August=COALESCE([8], 0),
September=COALESCE([9], 0),
October=COALESCE([10], 0),
November=COALESCE([11], 0),
December=COALESCE([12], 0)
```

FROM (

```
SELECT BranchID,MonthOfReservation,TotalRevenue
FROM ReservationRevenue
WHERE YearOfReservation = 2024
```

) AS SourceData

PIVOT (

```
SUM(TotalRevenue)
FOR MonthOfReservation IN ([1], [2], [3], [4], [5], [6], [7], [8], [9], [10], [11], [12])
```

) AS PivotedData;



פלט השאילתה:

	BranchID	January	February	March	April	May	June	July	August	September	October	November	December
1	1	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	7682.90	0.00
2	3	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1568.15
3	4	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	2552.66	0.00
4	24	0.00	0.00	0.00	0.00	0.00	3159.86	0.00	0.00	0.00	0.00	0.00	0.00
5	29	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	4110.82
6	30	3679....	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1628.19
7	31	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	11013.72	0.00	0.00	0.00
8	41	0.00	0.00	0.00	0.00	24...	0.00	0.00	0.00	0.00	0.00	0.00	0.00
9	45	0.00	0.00	920...	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
10	47	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	9169....	0.00	0.00
11	60	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1574.82
12	70	0.00	0.00	864...	0.00	93...	0.00	0.00	0.00	0.00	0.00	0.00	0.00
13	71	14526...	0.00	0.00	0.00	0.00	0.00	41...	0.00	0.00	0.00	0.00	0.00
14	76	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	7675.56	0.00	0.00	0.00
15	77	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	7058.99	0.00
16	86	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	2700.52	0.00
17	89	0.00	0.00	0.00	79...	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
18	90	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	4131.52	0.00	0.00	0.00
19	93	0.00	7457.99	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
20	94	0.00	6383.97	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
21	98	6276....	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
22	103	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	4970.00	0.00	0.00	0.00

פרויקט בסיסי נתונים חלק ב'	אוניברסיטת בן גוריון בנגב המחלקה להנדסת תעשייה וניהול
-------------------------------	--



טבלאות זמניות (Temporary Tables) :

טכניקת טבלאות זמניות ב-SQL היא כלי שימושי במיוחד לניהול תוצאות ביניים או נתונים שדורשים עיבוד חוזר לאורך חיי סשן או פרוצדורה. היא מאפשרת ליצור טבלאות שמבנה הנתונים שלהן, או התוכן שלהן, מוגדר באופן דינמי בזמן ריצה, כאשר הנתונים הללו זמינים רק למשך זמן מוגבל.

מתי כדאי להשתמש בטבלאות אלו:

- עיבוד שאילתות מורכבות
- שיפור ביצועים
- צמצום כפילויות
- שימוש חוזר בנתונים

הגיון עסקי: ממוצע של עלות ההשכרה לרכבים מסניף מסוים עבור כל התוצאות חיפוש.

נשתמש בטבלה זמנית כדי לאחסן את הנתונים הבסיסיים לעיבוד נוח.

נבחר טווח תאריכים מסוים וסניף מסוים ונבדוק מה הממוצע ההשכרות בתאריכים אלו. מידע זה יעזור לנו להבין אם יש חודש עם פחות השכרות אולי בגלל הממוצע הגבוה ונוכל לעשות הנחה מסוימת.

```
CREATE TABLE #TempSearchResults (
    BranchID INT,
    SearchIP VARCHAR(20),
    SearchDT DATETIME,
    CarID VARCHAR(17),
    PricePerDay DECIMAL(10, 2)
);
```

-- הכנסת נתונים לטבלה הזמנית --

```
INSERT INTO #TempSearchResults (BranchID, SearchIP, SearchDT, CarID, PricePerDay)
SELECT
```

```
    S.BranchT AS BranchID,
    S.SearchIP,
    S.SearchDT,
    R.CarID,
    R.PricePerDay
```

FROM

```
    SEARCHES AS S
```

```
    JOIN RESULTS R ON S.SearchIP = R.SearchIP AND S.SearchDT = R.SearchDT
```

WHERE

```
    S.BranchT = 200 -- סינון לפי סניף מס' 1
    AND S.StartDate >= '2024-01-01' AND S.FinishDate <= '2024-12-31';
```

-- חישוב עלות ממוצעת לרכבים בסניף זה --

```
SELECT
```

```
    BranchID,
```



```
AVG(PricePerDay) AS AvgPricePerDay
FROM
  #TempSearchResults
GROUP BY
  BranchID;

-- מחיקת הטבלה הזמנית בסיום
DROP TABLE IF EXISTS #TempSearchResults;
```

פלט השאילתה:

	BranchID	AvgPricePerDay
1	200	438.000000