דוח פעילות – מיני פרויקט בבסיסי נתונים

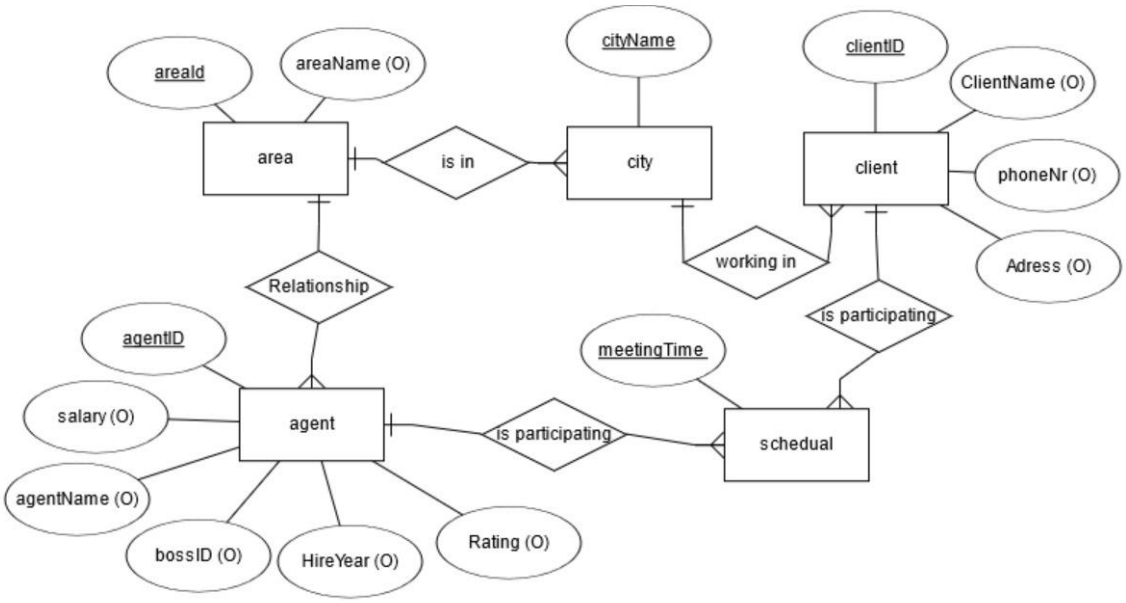
אחיה דנציגר :214092033
אלעד פישר:213924624

תוכן עניינים

Contents

תרגיל 0: היכרות עם הכלים

תרשים ERD:



קוד יצירת טבלאות:

```
CREATE TABLE area
(
  areaId INT NOT NULL,
  areaName INT,
  PRIMARY KEY (areaId)
);

CREATE TABLE agent
(
  agentID INT NOT NULL,
  agentName INT,
  Rating INT,
  HireYear INT,
  bossID INT,
  salary INT,
  areaId INT NOT NULL,
  PRIMARY KEY (agentID),
  FOREIGN KEY (areaId) REFERENCES area(areaId)
);
```

```
CREATE TABLE city
(
  cityName INT NOT NULL,
  areaId INT NOT NULL,
  PRIMARY KEY (cityName),
  FOREIGN KEY (areaId) REFERENCES area(areaId)
);

CREATE TABLE client
(
  clientID INT NOT NULL,
  ClientName INT,
  phoneNr INT,
  Adress INT,
  cityName INT NOT NULL,
  PRIMARY KEY (clientID),
  FOREIGN KEY (cityName) REFERENCES city(cityName)
);

CREATE TABLE schedual
(
  meetingTime_ INT NOT NULL,
  agentID INT NOT NULL,
  clientID INT NOT NULL,
  PRIMARY KEY (meetingTime_),
  FOREIGN KEY (agentID) REFERENCES agent(agentID),
  FOREIGN KEY (clientID) REFERENCES client(clientID)
);
```

אפיון:

אנחנו עשיהו את האפיון של האגף שאחראי על ההזמנות והביטולים של ההשכרות, הוא אחראי על שמירת כל ההשכרות והביטולים שהתבצעו, הוא גם אחראי על שמירת כל התשלומים שאנשים ביצעו.

בשביל מטרות אלה שומרים גם טבלאות קטנות השומרות את סוגי ההזמנות וסוגי ההוספות הקיימים ברשת.

טבלאות תחת אחריותנו:

הזמנות: **מזהה הזמנה**, תאריך התחלה, תאריך סיום, מזהה לקוח, מזהה מכונית, מזהה תמחור.

ביטולים: **מזהה ביטול**, תאריך ביטול, סיבה, מזהה הזמנה.

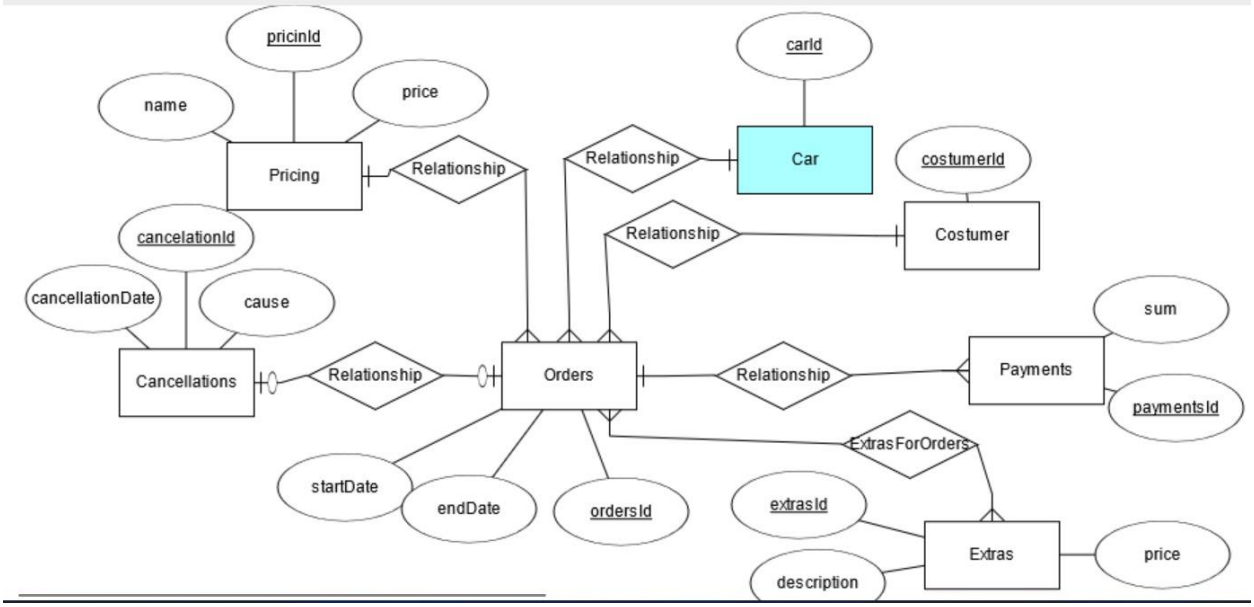
תשלומים: **מזהה תשלום**, סכום, מזהה הזמנה .

תמחורים: **מזהה תמחור**, שם, מחיר.

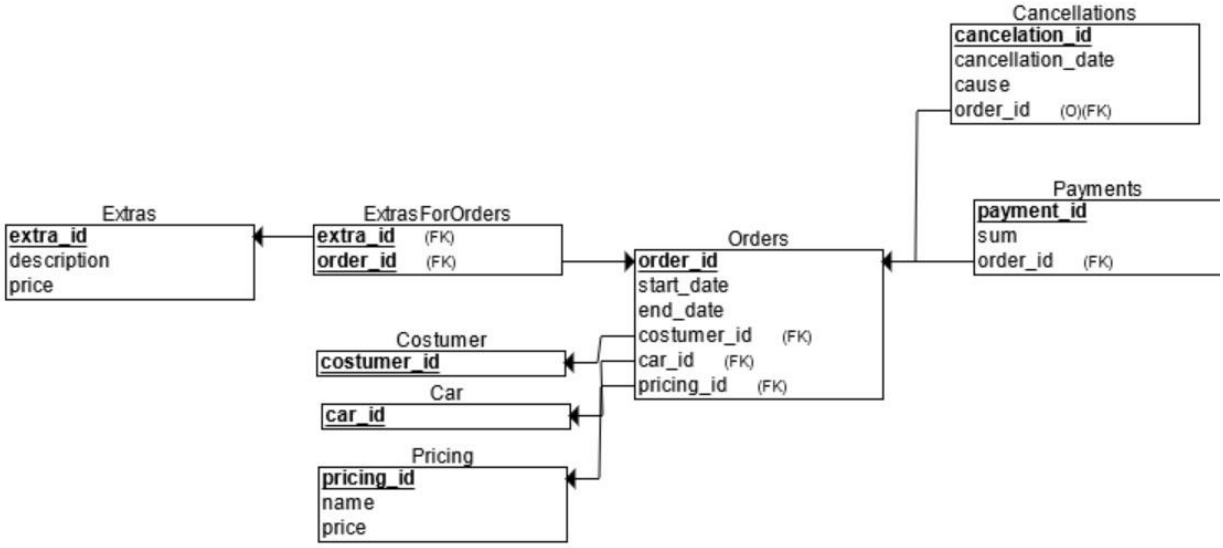
תוספות: **מזהה תוספת**, תיאור, מחיר.

תוספותלהזמנות: **מזהה הזמנה**, **מזהה תוספת**.

תרשים ERD :



אחיה דנציגר :214092033
אלעד פישר:213924624


Cancellations
cancelation_id
cancellation_date
cause
order_id (O)(FK)
Payments
payment_id
sum
order_id (FK)
Extras
extra_id
description
price
ExtrasForOrders
extra_id (FK)
order_id (FK)
Orders
order_id
start_date
end_date
costumer_id (FK)
car_id (FK)
pricing_id (FK)
Costumer
costumer_id
Car
car_id
Pricing
pricing_id
name
price

סקריפטים ליצירת טבלאות:

```sql
CREATE TABLE Pricing
(
  name VARCHAR(20) NOT NULL,
  price INT NOT NULL,
  pricing_id INT NOT NULL,
  PRIMARY KEY (pricing_id)
);

CREATE TABLE Costumers
(
  costumer_id INT NOT NULL,
  PRIMARY KEY (costumer_id)
);

CREATE TABLE Cars
(
  car_id INT NOT NULL,
  PRIMARY KEY (car_id)
);

CREATE TABLE Extras
(
  extra_id INT NOT NULL,
  description VARCHAR(80) NOT NULL,
  price INT NOT NULL,
  PRIMARY KEY (extra_id)
);
```

```sql
CREATE TABLE Orders
(
  order_id INT NOT NULL,
  start_date DATE NOT NULL,
  end_date DATE NOT NULL,
  payments INT NOT NULL,
  costumer_id INT NOT NULL,
  car_id INT NOT NULL,
  pricing_id INT NOT NULL,
  PRIMARY KEY (order_id),
  FOREIGN KEY (costumer_id) REFERENCES Costumers(costumer_id),
  FOREIGN KEY (car_id) REFERENCES Cars(car_id),
  FOREIGN KEY (pricing_id) REFERENCES Pricing(pricing_id)
);

CREATE TABLE Cancellations
(
  cancellation_date DATE NOT NULL,
  cause VARCHAR(80) NOT NULL,
  cancelation_id INT NOT NULL,
  order_id INT,
  PRIMARY KEY (cancelation_id),
  FOREIGN KEY (order_id) REFERENCES Orders(order_id)
);
```

```sql
CREATE TABLE Payments
(
  payment_id INT NOT NULL,
  sum INT NOT NULL,
  payement_month INT NOT NULL,
  order_id INT NOT NULL,
  PRIMARY KEY (payment_id),
  FOREIGN KEY (order_id) REFERENCES Orders(order_id)
);

CREATE TABLE ExtrasForOrders
(
  extra_id INT NOT NULL,
  order_id INT NOT NULL,
  PRIMARY KEY (extra_id, order_id),
  FOREIGN KEY (extra_id) REFERENCES Extras(extra_id),
  FOREIGN KEY (order_id) REFERENCES Orders(order_id)
);
```

את הסקריפטים האלו קיבלנו מתוך האתר erdPlus

יצירת מידע:

ביצירת מידע השתמשנו בסקריפט של python כי כשיצרנו מידע רצינו שיהיה התייחסויות בין הטבלאות השונות, למשל שאפשר לבטל הזמנות רק אחרי שהזמנו אותם, ושאי אפשר להזמין רכב כמה פעמים באותו יום.

את יצירת המידע התחלנו עם טבלה שמפרטת את מספר המופעים של טבלאות שאנחנו מקבלים:

```
number_of_instances = {"costumers": 5000, "car": 5000, "pricing": 3, "extras": 4}
```

את הטבלאות תמחורים ותוספות אנחנו מגדירים בצורה ידנית :

```
extras = pd.DataFrame(
    {"extra_id": [0, 1, 2, 3], "description": ["tablet", "booster_chair", "multimedia_system", "charger"],
     "price": [20, 10, 50, 5], "chance": [0.4, 0.1, 0.2, 0.5]})

pricing = pd.DataFrame({
    "pricing_id": [0, 1, 2], "name": ["regular", "new_driver", "summer"], "price": [150, 200, 300]
})
```

אחרי זה רצנו על כל המכניות ולכול מכונית הגדרנו מספר הזמנות כאשר אנחנו מתחילים מתאריך שהוא מספר ימים לפני התאריך המקורי, ואז רצים על הימים ומגדירים הזמנות שלאחריהם יש זמן כלשהו שבו המכונית לא מוזמנת עד שאנחנו מגיעים לתאריך הנוכחי, שם אנחנו מפסיקים את הריצה למכונית זו.

את כל השדות הנצרכים אנו מגדירים באופן רנדומלי לחלוטין למעט המספר המזהה שהוא מספר רץ, ותאריכי ההתחלה והסיום של ההזמנה שאת תאריך ההזמנה אנחנו מקבלים כסוף התקופה הנוכחית ומוסיפים לו עוד מספר ימים שלא יעלה על 10.

```
        df2 = pd.DataFrame(
            {"order_id": [num_orders + j], "start_date": [start_date], "end_date": [end_date], "car_id": [i],
             "costumer_id": [random.randint(0, number_of_instances["costumers"])],
             "pricing_id": [pricing_num],
             "payments": [payments_num], "sum_per_payment": [payment_sum/payments_num]})

        orders = pd.concat([orders, df2], ignore_index=True, axis=0)

        last_date = end_date

    num_orders = num_orders + j
```

בתוך הריצה אנחנו מחליטים גם באופן רנדומלי איזה הזמנות יהיו משויכות להזמנה בצורה רנדומלית, ואז אנחנו מכניסים את מה שיצא לנו לטבלה extrasForOrders.

```
num_orders = 0
for i in range(number_of_instances["car"]):
    last_date = datetime.date.today() - datetime.timedelta(days=90)

    j = 0
    while last_date < datetime.date.today():
        j += 1
        new_days = random.randint(1, 10)
        start_date = last_date + datetime.timedelta(days=random.randint(3, 30))
        end_date = start_date + datetime.timedelta(days=new_days)

        payments_num = random.randint(1, 10)
        pricing_num = random.randint(0, number_of_instances["pricing"] - 1)
        payment_sum = pricing["price"][pricing_num] * new_days
```

```
for index, extra in extras.iterrows():
    if random.random() < extra["chance"]:  # if this order was canceled
        df2 = pd.DataFrame({"order_id": [num_orders + j], "extra_id": [extra["extra_id"]]})
        extras_for_orders = pd.concat([extras_for_orders, df2], ignore_index=True, axis=0)
        payment_sum += extra["price"]
```

order_id	start_date	end_date	car_id	costumer_	pricing_id	payments
1	26/03/2022	28/03/2022	0	1200	0	10
2	05/04/2022	07/04/2022	0	3944	0	6
3	19/04/2022	28/04/2022	0	2547	0	3
4	25/05/2022	29/05/2022	0	3072	1	8
5	04/06/2022	12/06/2022	0	2506	2	5
6	14/03/2022	17/03/2022	1	2849	0	6
7	16/04/2022	23/04/2022	1	23	1	10
8	12/05/2022	20/05/2022	1	2381	0	2
9	09/06/2022	18/06/2022	1	2695	2	10
10	02/04/2022	05/04/2022	2	3071	0	6
11	21/04/2022	30/04/2022	2	2381	2	5
12	09/05/2022	12/05/2022	2	3376	2	4
13	29/05/2022	03/06/2022	2	3911	0	8
14	21/06/2022	22/06/2022	2	3468	0	9

אחרי זה אנחנו רצים על כל ההזמנות שהתבצעו, ולאחוז מסוים מהם אנחנו מגדירים שהוא בוטל ומגדירים מופע של ביטול הזמנה, עם מספר ההזמנה שלנו ומספר רץ חדש לביטולים , וסיבה שאנחנו מגרילים מרשימת הסיכות שיש לנו.

```
cancellation = pd.DataFrame(
    {"cancellation_id": [], "cancellation_date": [], "cause": [], "order_id": []})
num_cancellation = 0

for index, order in orders.iterrows():

    if random.random() < cancellation_percentage:  # if this order was canceled
        cancellation_date = f.date_between_dates(order["start_date"] - datetime.timedelta(days=7),
                                                 order["start_date"])

        df2 = pd.DataFrame(
            {"cancellation_id": [num_cancellation], "cancellation_date": [cancellation_date],
             "order_id": [order['order_id']],  "cause": [cause_list[random.randint(0, len(cause_list)) - 1]]})
        cancellation = pd.concat([cancellation, df2], ignore_index=True, axis=0)
        num_cancellation = num_cancellation + 1
```

cancellatic	cancellation_d	cause	order_id
0	07/05/2022	I killed my	25
1	11/03/2022	I was too l	33
2	20/03/2022	it costs toc	39
3	09/05/2022	I was too l	46
4	20/05/2022	I was too l	47
5	09/04/2022	an unexpe	51
6	19/03/2022	an unexpe	54
7	06/05/2022	I was too l	56
8	17/03/2022	the ac is n	64
9	22/04/2022	I killed my	80
10	13/06/2022	the bride (	130
11	12/03/2022	because	145
12	06/06/2022	an unexpe	148
13	25/05/2022	because	157
14	01/05/2022	the ac is n	182
15	14/05/2022	an unexpe	187
16	08/04/2022	I was too l	191
17	28/05/2022	the ac is n	198
18	23/04/2022	the bride (	205

אחרי זה אנחנו מגדירים את הטבלה של הביטולים כך:

אנחנו רצים על כל ההזמנות ושם אנחנו רצים על כל החודשים עד החודש הנוכחי ובכל חודש אנחנו משלמים את הסכום החודשי שצריכים לשלם על ההזמנה כסכום הכללי של ההזמנה חלקי מספר התשלומים בהזמנה:

```
payments = pd.DataFrame(
    {"payment_id": [], "payment_month": [], "order_id": [], "sum": []})
num_payments = 0

for index, order in orders.iterrows():
    current_month = order["start_date"].month + 1
    for _ in range(int(order["payments"])):
        if current_month > datetime.datetime.now().month:
            break
        df2 = pd.DataFrame(
            {"payment_id": [num_payments], "payment_month": [current_month],
             "order_id": [order['order_id']],  "sum": [order["sum_per_payment"]]})
        payments = pd.concat([payments, df2], ignore_index=True, axis=0)
        num_payments += 1
        current_month += 1
```

ayment_	payment_	order_id	sum
0	4	1	30
1	5	1	30
2	6	1	30
3	5	2	50.83333
4	6	2	50.83333
5	5	3	458.3333
6	6	3	458.3333
7	6	4	109.375
8	4	6	84.16667
9	5	6	84.16667
10	6	6	84.16667
11	5	7	140.5
12	6	7	140.5
13	6	8	630
14	5	10	79.16667
15	6	10	79.16667
16	5	11	540
17	6	11	540
18	6	12	225

ואחרי זה אנחנו מייצאים את כל הטבלאות לcsv ומשם העלינו אותם לerdPlus.

ביצירת מידע היינו תלויים בשני טבלאות השייכות לקבוצות אחרות:Customer, Car ומכיוון שידענו שכולם ישתמשו במזהים עם מספר רץ אנחנו התייחסנו לכל אחד מהם כמספר רץ בגבולות המספר שהגדרנו בהתחלה, כשאנחנו מתייחסים לכל שאר הנתונים שלהם כאל קופסה שחורה ולא נוגעים בהם בכלל.

את ההרצה הסופית על המידע הרצנו עם 5000 מכוניות ו-5000 אנשים ובסופו של דבר יצא לנו את מספר השורות הבא:

orders	23192
cancellations	2298
extras_for_orders	23191
payments	31980

שאילתות

שאילתה 1:

השאילתה בודקת כמה ימים כל אדם השכיר רכבים בסך הכל וממיינת את התוצאות לפי מספר הימים(כאשר רכבים כפולים ייחשבו כמו יומיים).

השאילתה בודקת כמה ימים כל השכרה לקחה ואז עושה group by לפי המזהה של המשתמש שהזמין.

```
select o.costumer_id, sum(daysRent.diff)
from orders o,(SELECT io.order_id,end_date - start_date as diff
from orders io
) daysRent
where o.order_id = daysRent.Order_Id
group by o.costumer_id
order by sum(daysRent.diff) desc;
```

הנה הדוגמה לתוצאה:

	COSTUMER_ID	SUM(DAYSRENT.DIFF)
1	1323	92
2	85	85
3	2940	84
4	656	83
5	1803	82
6	1359	81
7	195	81
8	858	81
9	2048	79
10	2387	79
11	240	79
12	2739	77
13	2985	77
14	584	76
15	1564	76

וההרצה לקחה בסך הכל 0.844 שורות.

שאילתה 2:

השאילתה הזו בודקת כמה צריך לשלם בכל הזמנה על התוספות שהזמינו

אנחנו בודקים את זה באמצעות בחירת כל התוספותלהזמנות לעשיית groupby עם הorder_id כך שאנחנו נקבל את הסכום של ההזמנות, ובנוסף אנחנו עושים איחוד לזה(שיש בו רק את ההזמנות שהיו להם תוספות) ביחד עם כל ההזמנות שלא היה להם תוספות בכלל.

```
select o.order_id, sum(e.price)
from orders o,extrasfororders efo,extras e
where o.order_id = efo.order_id  and efo.extra_id = e.extra_id
group by o.order_id
union
(select oLL.order_id, 0
from orders oLL,extrasfororders efo
minus
select distinct efo.order_id, 0
from extrasfororders efo)
```

	ORDER_ID	SUM(E.PRICE)
1	0	55
2	1	75
3	2	25
4	3	20
5	4	70
6	5	25
7	6	50
8	7	25
9	8	0
10	9	50
11	10	25
12	11	55
13	12	20
14	13	0
15	14	15
16	15	20

וההרצה לקחה 115.98 שניות להריץ.

אחיה דנציגר :214092033
אלעד פישר:213924624

שאילתה 3:

השאילתה מחזירה הזמנות שהתבטלו בגלל אחת משלוש סיבות וכאשר מספר האקסטרות של ההזמנה גדול מ-2 וגם מספר הימים של ההזמנה קטן מ-4, כלומר אנחנו מחפשים הזמנות רווחיות במיוחד.

```
(select o.order_id,T.extrasNumber, o.end_date - o.start_date
from orders o, (select  P.ORDER_ID , count(*) as extrasNumber
             from Extrasfororders P
             group by P.order_id ) T
where exists (select * from cancellations R where
 (R.CAUSE = 'my kids are annoying'
or R.CAUSE = 'it costs too much for us to justify it'
or R.CAUSE = 'I moved to the competitor')
 and R.ORDER_ID = o.order_id )
 and T.extrasNumber > 2 and T.order_id = o.order_id
)
minus
(
SELECT
  o.order_id, T.extrasNumber, o.end_date - o.start_date
FROM orders o,(select  P.ORDER_ID , count(*) as extrasNumber
             from Extrasfororders P
             group by P.order_id ) T
where o.end_date - o.start_date > 5)
```

זו דוגמה לתוצאות שקיבלנו:

	ORDER_ID	EXTRASNUMBER	DURATION
1	380	3	3
2	467	3	4
3	1406	3	2
4	1440	3	2
5	2542	3	4
6	2588	3	4
7	2639	3	3
8	2944	3	2
9	3884	3	3
10	3993	3	4
11	5835	3	1
12	6226	3	5
13	6238	3	3
14	6349	3	4
15	6656	3	4

ובסך הכל ההרצה לקחה 86.57 שניות.

שאילתה 4:

בודקים לכל לקוח את כמות הכסף שהוא חייב לשלם לחברה אחרי כל התשלומים שהוא ביצע, את זה עושים כשבודקים לכל הזמנה שלו כמה עוד תשלומים נשארו ומכפילים את זה בסכום הכללי של כל ההזמנה ועושים על זה group by על השם של הלקוח

```
select orders.costumer_id , sum(pay.price / orders.payments * paymentsLeft.Diff) as sumLeft
from orders,pay,
     (select o.payments - numpayment.temp  as diff, o.order_id
      from orders o,(
      select payments.order_id, count(*) as temp
      from payments
      group by payments.order_id ) numPayment
      where o.order_id = numPayment.order_id  ) paymentsLeft
where orders.order_id = paymentsLeft.order_id and pay.order_id = orders.order_id
group by orders.costumer_id
order by orders.costumer_id
```

	COSTUMER_ID	SUMLEFT
1	0	7167.11904761905
2	1	2482.5
3	2	6442.5
4	3	5376.0119047619
5	4	1782.5
6	5	3734.5
7	6	3000.5
8	7	6092.61904761905
9	8	2980.22222222222
10	9	3096.77777777778
11	10	0
12	11	875
13	12	5683.92857142857
14	13	8257.97222222222
15	14	2146.66666666667
16	15	3604.04761904762
17	16	5268.9880952381
18	17	3849.86904761905
19	18	4473.54166666667
20	19	4242.44444444444

20 rows selected in 116.607 seconds (more...)

שאילתה 5:

בשאילתה הזו אנחנו בודקים כמה כסף צריך לשלם כל הזמנה לפי כל הפרמטרים האחרים שקיבלנו.

```
SELECT orders.order_id,(end_date - start_date) * pricing.price + sumExtras.price price
from orders,pricing,sumExtras
where orders.pricing_id = pricing.pricing_id and sumExtras.order_id = orders.order_id
order by orders.order_id;
```

	ORDER_ID	PRICE
1	0	1405
2	1	825
3	2	2425
4	3	1420
5	4	2170
6	5	325
7	6	2450
8	7	925
9	8	800
10	9	650
11	10	2125
12	11	655
13	12	1520
14	13	600
15	14	2115
16	15	2720
17	16	1070
18	17	200
19	18	1205
20	19	1225
21	20	1205
22	21	905
23	22	1205
24	23	2725
25	24	1855
26	25	1105

& 4:27 1:56 elafishe@labdbwin [23:20:27] 26 rows selected in 116.794 seconds (more...)

שאילתה 6:

```
SELECT
  order_id,
  end_date - start_date   daysRent
FROM orders
where end_date - start_date < 3
```

השאילתה מחשבת את הטבלה של ההשכרות הגרועות, כולמר השכרות שהיו קצרות והן פחות רווחיות מבחינת הימים שהושכרו לעבודה הנדרשת מאשני המכירות, המזכירות, ואנשי הצוות.

דוגמת הרצה של השאילתה:

	ORDER_ID	DAYSRENT
1	3	1
2	9	2
3	17	1
4	29	2
5	30	1
6	33	1
7	40	2
8	41	1
9	50	1
10	57	1
11	62	2
12	65	1
13	70	1
14	79	2
15	90	1
16	93	1
17	103	1
18	106	1
19	109	1
20	110	1
21	111	2
22	112	1
23	116	1
24	123	1
25	124	1
26	127	2
27	134	1
28	136	1
29	138	2
30	142	1

שאילתה 7:

```
select cars.car_id , extras.extra_id,count(*)
from cars,orders,Extrasfororders,extras
where cars.car_id = orders.car_id and
orders.order_id = extrasfororders.order_id and
extrasfororders.extra_id = extras.extra_id
group by cars.car_id, extras.extra_id
having count(*) > 4
order by Cars.Car_Id
```

השאילתה מחשבת את כמות התוספות שכל רכב קיבל לשימוש לאורך תקופת ההשכרות שלו. שאילתה זו חשובה כי אולי יש רכבים מסוימים שמזמינים אליהם תוספות מיוחדות.

דוגמת הרצה:

	CAR_ID	EXTRA_ID	COUNT(*)
1	0	0	6
2	0	2	5
3	0	3	5
4	1	0	5
5	1	3	5
6	3	3	9
7	16	0	5
8	16	3	7
9	17	3	5
10	22	0	5
11	30	0	5
12	30	3	6
13	31	3	5
14	36	0	5
15	39	3	6
16	46	0	5
17	52	3	5
18	56	0	5
19	57	3	6
20	58	0	7
21	60	0	6
22	60	3	5
23	62	3	6
24	63	3	5
25	64	3	5
26	68	3	5
27	69	3	7
28	70	0	7
29	73	0	5

שאילתה 8:

```
select o.order_id, o.start_date,
 o.end_date, c.car_id, cu.costumer_id
from orders o, CARS C ,Costumers CU
where o.end_date - current_Date >= 0 and
 current_date - o.start_date >= 0 and
 c.car_id = o.car_id and
 o.costumer_id = CU.costumer_id
order by o.order_id
```

השאילתה שלנו מחשבת את כל ההזמנות שמתקיימות היום ומביאה טבלה עם מספר הלקוח מספר הרכב ותאריך ההתחלה והסוף וכמובן מספר ההזמנה. השאילתה הזאת יעיל לנו כי זה מידע מעניין איזה השכרות קורות היום.

דוגמת הרצה:

	ORDER_ID	START_DATE	END_DATE	CAR_ID	COSTUMER_ID
1	9	06/06/2022	08/06/2022	0	2859
2	277	05/06/2022	08/06/2022	46	298
3	510	05/06/2022	11/06/2022	79	2049
4	1375	03/06/2022	08/06/2022	211	123
5	3341	29/05/2022	07/06/2022	514	2216
6	4114	30/05/2022	08/06/2022	631	901
7	4130	31/05/2022	09/06/2022	633	1042
8	5287	05/06/2022	11/06/2022	804	1578
9	7060	01/06/2022	09/06/2022	1075	911
10	7817	02/06/2022	11/06/2022	1196	1443
11	7916	31/05/2022	07/06/2022	1210	1281
12	9145	29/05/2022	07/06/2022	1397	1551
13	9490	06/06/2022	14/06/2022	1447	1259
14	9862	03/06/2022	09/06/2022	1499	2457
15	9988	30/05/2022	08/06/2022	1519	2604
16	12472	30/05/2022	07/06/2022	1891	2917
17	13313	03/06/2022	12/06/2022	2019	117
18	14122	31/05/2022	08/06/2022	2145	789
19	15209	02/06/2022	10/06/2022	2307	1374
20	15874	05/06/2022	09/06/2022	2405	603
21	18422	30/05/2022	07/06/2022	2809	2904
22	19387	02/06/2022	07/06/2022	2968	983

אינטגרציה

עשינו אינטגרציה עם חיים ולמעשה הבאנו לו את הטבלאות שלנו, למשל:

למעשה יש כמה מקבוצה שביקשו את הטבלאות שלנו משום שאנחנו יחסית עם הרבה ישויות ומה שיש לנו משמעותי יחסית. אבל מחקנו את הטבלאות כדי לעדכן עוד מידע וכל מי שביקש ממנו הרשאה זה נמחק לו.

Grantee	Select	Insert	Update	Delete	References	Alter	Index	Read	Debug
miletzky	Yes								

את הישויות שלנו לקחנו ליוזר

Dastraus

אחיה דנציגר :214092033
אלעד פישר:213924624

שאילתה 1:

```
select C.LICENSE_PLATE,O.ORDER_ID, CU.NAME, CU.CUSTOMER_ID,C.CAR_MODEL
from dastraus.cars C, orders O,dastraus.customers CU
where C.LICENSE_PLATE = O.Car_ID and CU.CUSTOMER_ID = o.costumer_id and C.ACTIVE = 0
order by C.LICENSE_PLATE
```

השאילתה הזאת בעצם מחפשת את כל הרכבים הלא פעילים שיש לנו בצי ואת ההשכרות שהרכב שוייך אליהן, אנחנו רוצים לדעת מה ההשכרות שצריך להחליף להן לרכב פעיל, כולל את מודל הרכב ושאר הפרטים על מנת שנוכל למצוא רכב טוב ככל האפשר במקום

דוגמת הרצה:

	LICENSE_PLATE	ORDER_ID	NAME	CUSTOMER_ID	CAR_MODEL
1	1	14	Leo Mars	2375	ML-023
2	1	18	Terrence Lonsdale	2530	ML-023
3	1	11	CeCe Nash	1470	ML-023
4	1	13	Madeline Walken	948	ML-023
5	1	12	Terri Posey	619	ML-023
6	1	15	Walter Breslin	670	ML-023
7	1	10	Debra Clooney	233	ML-023
8	1	16	Vondie Gaines	33	ML-023
9	1	19	Remy Hershey	565	ML-023
10	1	17	Scott Perry	658	ML-023
11	2	21	Mel Dutton	2212	RD-031
12	2	20	Frederic England	301	RD-031
13	2	22	Renee Masur	176	RD-031
14	4	35	Jena Hauer	2223	CMA-154
15	4	34	Gwyneth Harrelson	2601	CMA-154
16	4	33	Burt Carradine	1894	CMA-154
17	6	45	Swoosie Sample	2873	CMA-161
18	6	43	Russell Rio	1635	CMA-161
19	6	48	Trey Chao	2496	CMA-161
20	6	44	Bo Janney	1742	CMA-161
21	6	46	Wayne Flanagan	1231	CMA-161
22	6	47	Sam Frost	472	CMA-161
23	6	42	Julia Stiers	1146	CMA-161
24	7	49	Lisa Nugent	2998	CP-035
25	7	51	Ivan Rebhorn	1223	CP-035
26	7	52	Alana Warren	1982	CP-035
27	7	50	Bryan Hersh	2296	CP-035
28	9	57	Kitty McBride	1119	ABB-060
29	9	59	Jamie Belles	789	ABB-060
30	9	61	Devon Wiedlin	26	ABB-060
31	9	58	Joanna Haslam	196	ABB-060

שאילתה 2:

```
select CU.CUSTOMER_ID,CU.NAME,count(*)
from dastraus.customers CU, dastraus.cars C, orders o
where  C.LICENSE_PLATE = O.Car_ID and
 CU.CUSTOMER_ID = o.costumer_id and
 C.MANUFACTURING_DATE < CU.DATE_OF_BIRTH
group by CU.CUSTOMER_ID,CU.NAME
having count(*)  >  4
order by count(*) desc
```

כאן אנחנו מחפשים את כל האנשים שמשכירים רכב שיותר מבוגר מהם, אם הם השכירו מעל 4 פעמים, כלומר 5 פעמים או יותר את הרכבים שיותר מבוגרים מהם...

דוגמת הרצה:

	CUSTOMER_ID	NAME	COUNT(*)
1	2940	Catherine Cleese	17
2	1359	Hugo Buckingham	14
3	2116	Jarvis Hidalgo	13
4	1323	Janeane Hoffman	13
5	2895	Diamond Spacek	13
6	1963	Mena Ferrer	12
7	209	Mika Remar	12
8	2269	Gordon Preston	12
9	1947	Suzy Lizzy	12
10	2759	Mika Ripley	12
11	419	Mickey Vance	11
12	1413	Azucar Sossamon	11
13	775	Rhett Goldwyn	11
14	901	Eliza Burstyn	11
15	1625	Casey Clarkson	11
16	2922	Jeroen Arthur	11
17	85	Dar Quinones	11
18	591	Maury Addy	11
19	505	Jean-Claude Tambor	11
20	931	Terence Cummings	11
21	2618	Julianna Ontiveros	11
22	59	Kitty Affleck	11
23	2918	Whoopi Sutherland	11
24	448	Jann Loggins	11
25	2631	Garth Crudup	11
26	2425	Debbie Diaz	11
27	1089	Rodney Janssen	11
28	1568	Martin Head	11
29	2019	Wendy Dench	11

שאילתה 3:

```
select EXTRACT (YEAR from C.MANUFACTURING_DATE), count(*)
from dastraus.cars C,orders o,cancellations ca
where o.order_id = ca.cancelation_id and c.license_plate = o.car_id
group by EXTRACT (YEAR from C.MANUFACTURING_DATE)
order by EXTRACT (YEAR from C.MANUFACTURING_DATE) desc
```

השכרה על פי שנים:

כאן אנחנו נעשה שאילתה לדעת את מספר ההשכרות שהתרחשו אצלנו לפי שנת הייצור של הרכב, זהו מידע מעניין כי אני רוצה לדעת את הפופולריות של רכבים שהשכרנו בחברה שלנו.

דוגמת הרצה:

	MANUFACTORED_YEAR	COUNT(*)
1	2004	71
2	2003	96
3	2002	26
4	2001	54
5	2000	124
6	1999	122
7	1998	55
8	1997	70
9	1996	69
10	1995	55
11	1994	74
12	1993	51
13	1992	79
14	1991	97
15	1990	75
16	1989	29
17	1988	12
18	1987	75
19	1986	75
20	1985	45
21	1984	62
22	1983	85
23	1982	69
24	1981	69
25	1980	94
26	1979	94
27	1978	87
28	1977	70
29	1976	65
30	1975	100

אינדקסים:

נריץ 3 אינדקסים שיצרנו:

```
create index indexOrdersStartDate on
orders (start_date);

create index cancellaionsIndex on
cancellations (cause);

create index ExtrasfororordersIndex on
Extrasfororders (extra_id);
```

הקוד שלנו יוצר 3 אינדקסים:

אחד על התאריך התחלה של ההשכרה מה שמשפר כי יש לנו יחסית הרבה השכרות כל תאריך

אחד על סיבת הביטול כי יש לנו כ15-20 סיבות כרגע ועוד סיבות לעתיד, אך רוב הסיבות אמורות להיות דומות כך שיהיה לנו יותר קל לחפש מידע שקשור לביטול

ועוד אינדקס אחד על מספר התוספת בטבלת ההוספות.

שינוי מספר 1:

```
select *
from cancellations R
where R.CAUSE = 'my kids are annoying'
```

```
select *
from cancellations R
where R.CAUSE = 'my kids are annoying'
```

29:1 elafishe@labdbwin [19:48:40] 32 rows selected in 0.033 seconds (more...)

אחרי שניצור את האינדקס שלנו ונריץ את השאילתה נקבל

```
select *
from cancellations R
where R.CAUSE = 'my kids are annoying'
```

	CANCELLATION_DATE	CAUSE	CANCELATION_ID	ORDER_ID
1	20/05/2022	my kids are annoying	3	7
2	11/03/2022	my kids are annoying	8	51
3	26/03/2022	my kids are annoying	25	127

29:1 elafishe@labdbwin [19:50:48] 18 rows selected in 0.086 seconds (more...)

הבדל זה מאוד משמעותי.

הזמן הראשון היה 0.033 שניות ואחרי יצירת האינדקסים זה לקח 0.086 שניות, הבדל זה משמעותי הוא הבדל של 160% יותר זמן לרעה, הבדל זה נובע כי הוא מחפש את השורות על פי האינדקס וזה לוקח יותר זמן מסידור פשוט ומעבר על השאילתות

שינוי מספר 2:

הרצנו את השאילתה שמחפשת הזמנות חופפות כאשר אחת ההזמנות התבטלה והשניה לא והביטול היה בגלל סיבה ספציפית, זה לקח לנו 0.07 שניות לפני השינוי ו0.1123 אחרי. זה מייצג שינוי של פי 175% לרעה, שוב, בגלל שאנחנו מחפשים במקום ספציפי זה רע כי מספיק לנו לעבור שורה שורה וזה יוצא מהיר יותר ככה

הזמן לפני יצירת האינדקסים:

```
select o1.order_id, o2.order_id, o1.costumer_id
from orders o1,orders o2
where exists (select * from cancellations R where R.CAUSE = 'my kids are annoying' and R.ORDER_ID = o1.order_id ) and not exists (select * from cancellations R where R.CAUSE = 'my kids
```

	ORDER_ID	ORDER_ID	COSTUMER_ID
1	18866	11341	2433
2	16202	7434	2901
3	8591	5248	321
4	10120	2842	2948
5	19505	13642	24
6	10516	1805	939
7	15035	7983	2444
8	19427	17769	13
9	10327	1875	69
10	15271	15149	574
11	4022	2010	2040

37:1 elafishe@labdbwin [19:57:33] 23 rows selected in 0.070 seconds (more...)

הזמן אחרי יצירת האינדקסים:

```
select o1.order_id, o2.order_id, o1.costumer_id
from orders o1,orders o2
where exists (select * from cancellations R where R.CAUSE = 'my kids are annoyin
```

	ORDER_ID	ORDER_ID	COSTUMER_ID
1	18866	11341	2433
2	16202	7434	2901
3	8591	5248	321
4	10120	2842	2948
5	19505	13642	24
6	10516	1805	939
7	15035	7983	2444
8	19427	17769	13
9	10327	1875	69
10	15271	15149	574
11	4022	2010	2040
12	18235	10143	1703
13	14831	9492	2
14	13196	7250	2058
15	10173	3543	2471
16	19427	10209	13
17	18673	8638	2377
18	17902	11791	2193

37:1 elafishe@labdbwin [19:58:35] 18 rows selected in 0.123 seconds (more...)

שיפור 3:

הרצנו את השאילתה של ההזמנות עם מספרי האקסטרות עבור הזמנות שבוטלו מסיבות מסוימות וגם השכיר את הרכב ל5 ימים או פחות:

```
(select o.order_id,T.extrasNumber
from orders o, (select  P.ORDER_ID , count(*) as extrasNumber
             from Extrasfororders P
             group by P.order_id ) T
where exists (select * from cancellations R where
 (R.CAUSE = 'my kids are annoying'
or R.CAUSE = 'it costs too much for us to justify it'
or R.CAUSE = 'I moved to the competitor')
 and R.ORDER_ID = o.order_id )
 and T.extrasNumber > 2 and T.order_id = o.order_id
)
minus
(
SELECT
  o.order_id, T.extrasNumber
FROM orders o,(select  P.ORDER_ID , count(*) as extrasNumber
             from Extrasfororders P
             group by P.order_id ) T
where o.end_date - o.start_date > 5)
```

	ORDER_ID	EXTRASNUMBER
1	380	3
2	467	3
3	1406	3
4	1440	3
5	2542	3
6	2588	3
7	2639	3
8	2944	3

12:1 0:58 elafishe@labdbwin [22:03:11] 40 rows selected in 58.552 seconds

כאשר אנחנו מריצים את זה בלי אינדקסים אנחנו מקבלים זמן 58.5 שניות זמן ריצה.

אך כאשר אנחנו מריצים עם אינדקסים הזמן משתפר פלאים ל57.5 שניות ככה

```
(select o.order_id,T.extrasNumber
from orders o, (select  P.ORDER_ID , count(*) as extrasNumber
             from Extrasfororders P
             group by P.order_id ) T
where exists (select * from cancellations R where
 (R.CAUSE = 'my kids are annoying'
or R.CAUSE = 'it costs too much for us to justify it'
or R.CAUSE = 'I moved to the competitor')
 and R.ORDER_ID = o.order_id )
 and T.extrasNumber > 2 and T.order_id = o.order_id
)
minus
(
SELECT
  o.order_id, T.extrasNumber
FROM orders o,(select  P.ORDER_ID , count(*) as extrasNumber
             from Extrasfororders P
             group by P.order_id ) T
where o.end_date - o.start_date > 5)
```

	ORDER_ID	EXTRASNUMBER
1	380	3

17:1 0:57 elafishe@labdbwin [22:11:48] 18 rows selected in 57.450 seconds (more...)

שיפור זה משפר לנו בשניה ו5 מאיות את ההרצה שלנו וזה מתבטא גם בשיפור של 1.2% השיפור קורה בעיקר כי כנראה שבסקאלה של הנתונים הזה אנחנו גורמים לשיפור בקביעה של סיבות הביטול שאנחנו יכולים להתמקד בסיבה אחת ספציפית ולא לבדוק את כל הסיבות כמו בחיסור בין הטבלאות.

View

View1:

View ראשון שעשינו הוא:

```
create view today as
select *
from orders o,DASTRAUS.CARS C,DASTRAUS.CUSTOMERS CU
where o.end_date - current_Date >= 0 and
 current_date - o.start_date >= 0 and
 c.license_plate = o.car_id and o.costumer_id = CU.CUSTOMER_ID
order by o.order_id
```

התפקיד שלו ליצור את הטבלה של ההזמנות היומיות, אנחנו למעשה רוצים לייצר את זה פעם ביום ומקסימום עוד פעם אם מתווספת הזמנה יומית, ולכן אנחנו נחשב את כל ההזמנות שמתרחשות היום בפעם אחת.

view2:

```
create view SumExtras as
(select o.order_id, sum(e.price) as price
from orders o,extrasfororders efo,extras e
where o.order_id = efo.order_id  and efo.extra_id = e.extra_id
group by o.order_id
union
(select oLL.order_id, 0
from orders oLL,extrasfororders efo
minus
select distinct efo.order_id, 0
from extrasfororders efo));
```

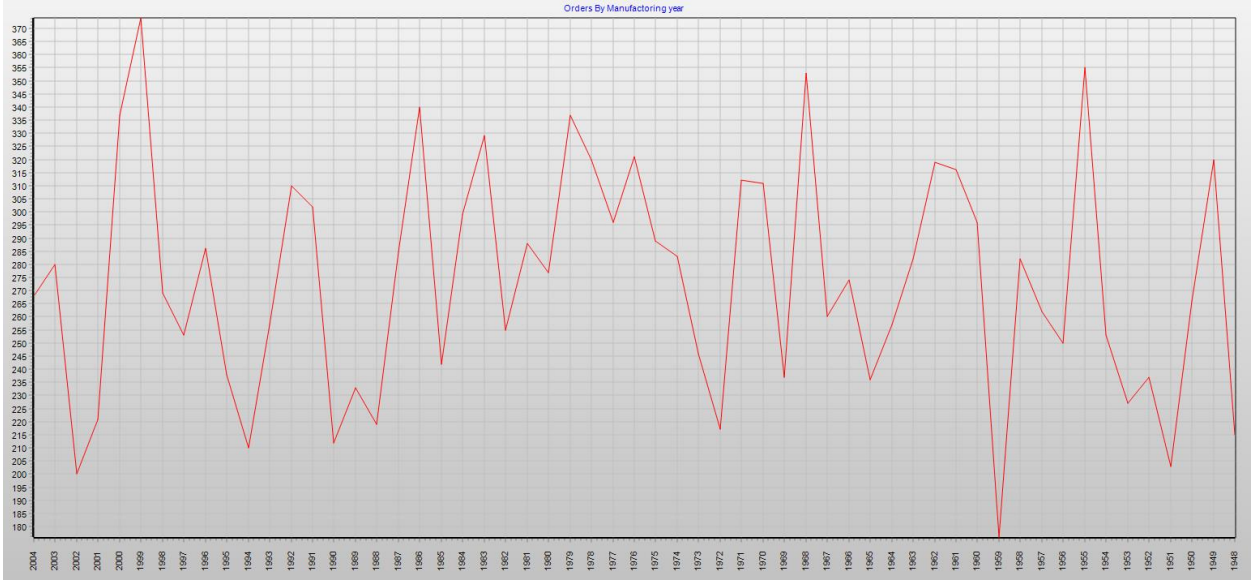
זה מייצר את הטבלה של התוספות מחיר של התוספות שכל השכרה לוקחת, למעשה אנחנו כנראה לא נרצה לעשות את החישוב יותר מכמה פעמים ביום ולמעשה זה טבלה יעילה כמו שאפשר היה להשתמש בה ברוב השאילתות שצריכות את המידע על תשלומים ואפשר לשלב אותה במגוון שאילתות כמו:

כמה צריך לשלם על תוספות ללקוח מסוים, כמה יהיה התשלום החודשי של הלקוחות, וכו'

גרפים

גרף 1:

גרף אחד שעשינו היה מספר השכרות על פי שנת ייצור של רכב, למעשה זהו גרף חשוב ביותר כי אפשר ללמוד מתוכו על העדפות של המשכירים לגבי מודלים מסוימים של שנת ייצור של רכבים, למשל אפשר לראות שהרכב הכי חדש הוא מ2004 ולמשל היו שנים מאוד מוצלחות ושנים שלא אהבו במיוחד את הרכבים.

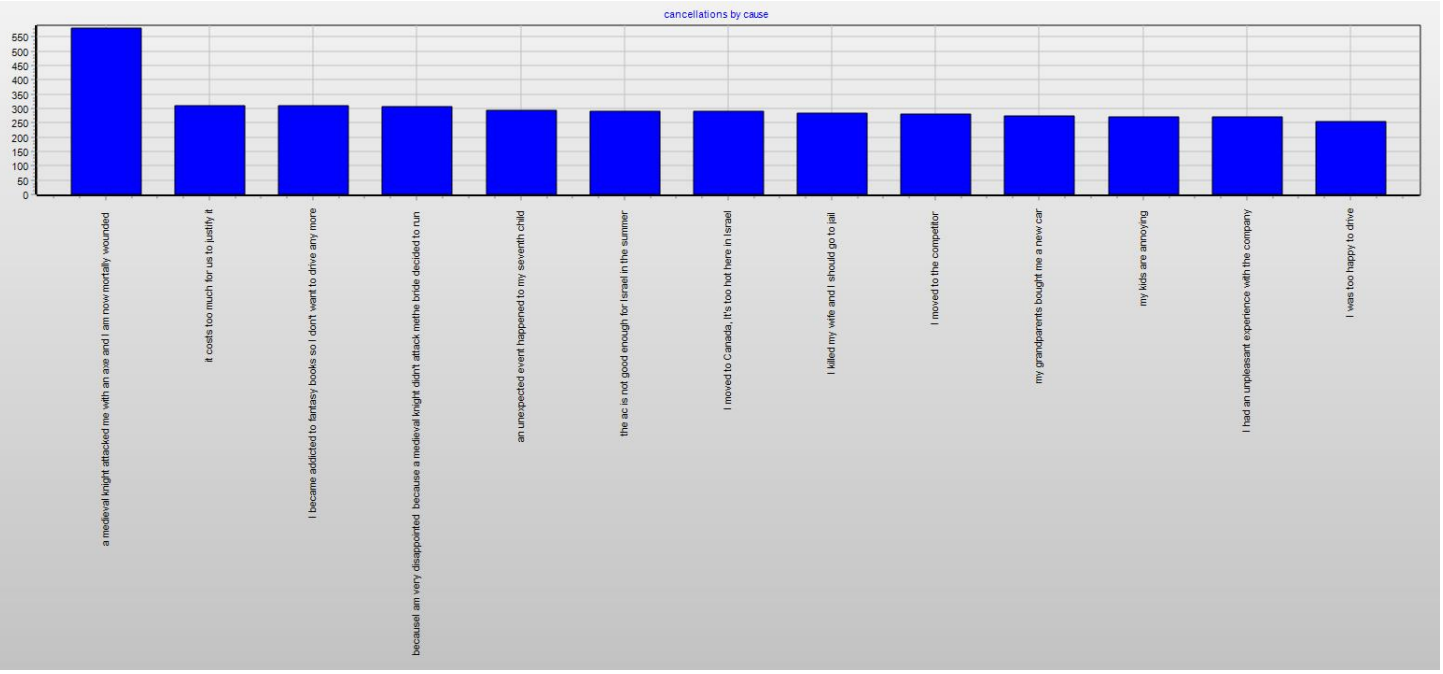


השאילתה שיוצרת את הגרף היא:

```
select EXTRACT (YEAR from C.MANUFACTURING_DATE), count(*)
from dastraus.cars C,orders o
where not exists (select * from Cancellations Can where o.order_id = Can.Order_Id) and c.license_plate = o.car_id
group by EXTRACT (YEAR from C.MANUFACTURING_DATE)
order by EXTRACT (YEAR from C.MANUFACTURING_DATE) desc
```

גרף 2:

מספר ביטולי השכרות לפי כל סיבה, הגרף הזה יעיל מאוד כדי לנתח ויזואלית את הסיבות לביטולים.



למשל אפשר ללמוד על פי הגרף הזה שיש לנו הטיה מיוחדת לסיבה שאנשים ביטלו בגלל שאביר מימי הביניים תקף אותם ועכשיו הם פצועים אנושות.

הקוד שיוצר את הגרף הוא:

```
select c.cause, count(*)
from cancellations c
group by c.cause
order by count(*) desc
```

פונקציות

פונקציה מספר 1: מחיר עבור הזמנה

מחזירה את המחיר של התוספות עבור הזמנה מסוימת. הפונקציה הזאת יעילה עבור מקרים בהם נרצה לבדוק את המחיר עבור השכרה מסוימת או מספר השכרות לפי מספרי הזמנה ספציפיים.

```
create or replace function getExtrasPriceOfOrder( orederId number) return
is
  res number;

  cursor extrasPerPeople is(
  select sum(e.price) as priceOfExtras
  from extrasfororders efo,extras e
  where orederId = efo.order_id  and efo.extra_id = e.extra_id
  );

  rowInTable extrasPerPeople%rowtype;
  total number;

begin
  total := 0;
  open extrasPerPeople;
  if(extrasPerPeople%Notfound) then
   res := 0;
  else
   loop
     fetch  extrasPerPeople into rowIntable;
     exit when extrasPerPeople%Notfound;
     total := total + rowIntable.priceOfExtras;
   end loop;
  end if;
  res := total;
  return (res);
end getExtrasPriceOfOrder;
```

בגדול הפונקציה מייצרת טבלה של כל התוספות שאדם מסוים לקח ואז היא מחברת את כל המחירים של ההוספות ומחזירה את המחיר הכולל, החוזק של הפונקציה בא לידי ביטוי כאשר יש הזמנה שיש לה 0 מחיר לתוספים ואז זה למעשה יחזיר 0 וזה יהיה יותר פשוט מהשאילתה המסובכת שעשינו

דוגמת הרצה להרצת הפונקציה :

```
declare
  result number;
  orderNumber number := 2;
begin
  result := getExtrasPriceOfOrder(orderNumber);
  dbms_output.put_line('sum of extras for order '||
  to_char(orderNumber,'999999') ||' is: '|| to_char(result,'999.99'));
end;
```

דוגמה לפלט של הקריאה הזאת:

```
sum of extras for order       2 is:   30.00
```


אחיה דנציגר :214092033
אלעד פישר:213924624

פונקציה 2: בדיקת זמינות של מכונית

הפונקציה בודקת אם מכונית מסוימת פנויה בטווח מסוים של ימים. הפונקציה שימושית בעיקר כדי לראות אם רכב מתאים להזמנה או כדי לחפש רכב להזמנה עתידית:

```
create or replace function isCarFree(carId integer, orderStart date, orderFinish date) return boolean
is
res boolean;

emptyDay number;

begin

  select count(*) into emptyDay from orders o
  where o.car_id = carId and not((o.start_date - orderFinish >= 0) or (orderStart - o.end_date >= 0));

  if orderStart > orderFinish then
    dbms_output.put_line('invlid dates');
  end if;

  if (emptyDay = 0) then
    res := true;
    dbms_output.put_line('empty');
  else
    dbms_output.put_line('full');
    res := false;
  end if;

  return (res);
end iscarfree;
```

ברעיון הפונקציה מייצרת טבלה של כל ההזמנות החופפות בתאריך של אותו רכב ומכניסה את זה למשתנה מסוים שאם הוא שווה 0 אז הרכב פנוי. אם התאריכים לא טובים היא תדפיס שזה תאריך לא מתאים ואחרת היא תחזיר את הרכב פנוי או לא ותדפיס את התוצאה.

דוגמת הרצה של הפונקציה עם הרכב מספר 3:

```
declare
c number := 3;
startD date := to_date('05/03/2022','dd/mm/yyyy');
endD date := to_date('06/03/2022','dd/mm/yyyy');
res boolean;
begin
  res := isCarFree (c,startD,endD);

  if res = true then
    dbms_output.put_line('empty dates');
   else
     dbms_output.put_line('full dates');
 end if;
end;
```

והפלט יהיה:

```
empty
empty dates
```

ולכן הרכב פנוי, אך נקרא לפונקציה עם טווח תאריכים אחר שהרכב יהיה תפוס בו, למשל:

```
--call for is car free function

declare
c number := 3;
startD date := to_date('05/03/2022','dd/mm/yyyy');
endD date := to_date('06/04/2022','dd/mm/yyyy');
res boolean;
begin
  res := isCarFree (c,startD,endD);

  if res = true then
    dbms_output.put_line('empty dates');
   else
     dbms_output.put_line('full dates');
 end if;
end;
```

אזי הפלט יהיה:

```
full
full dates
```

פרוצדורה 1: בדיקת ביטול רכב

פרוצדורה מספר 1 שלנו בודקת ומדפיסה האם רכב מסוים התבטל לפני תאריך מסוים. זה יכול להיות שימושי למשל כדי לבדוק את כל הרכבים שהתבטלו שבוע או יותר לפני ההשכרה או באמצע ההשכרה.

```
create or replace procedure isCanncelled( OrderNumber integer , Cancelldate Date)
is
--x number:=0;

cancelled boolean := false;

cursor CancellationsDates is
       select c.order_id as OrderId, c.cancellation_date as WhenCancelled
       from cancellations c
       order by c.order_id;
sl CancellationsDates%Rowtype;
begin
  open CancellationsDates;
  if(CancellationsDates%Notfound) then

  dbms_output.put_line('No Found cancellations');

  else
    --dbms_output.put_line('orders That Cancelled before ' || TO_CHAR(Cancelldate, 'yyyy/mm/dd'));

    loop
      fetch CancellationsDates into sl;
      exit when CancellationsDates%notfound;

      if (CancellDate - sl.WhenCancelled >= 0 and sl.OrderId = OrderNumber ) then
        cancelled := true;
        dbms_output.put_line('Order '||to_char(sl.orderID,'999999')||' cancelled');
      end if;

    end loop;

    if( cancelled = false) then
      dbms_output.put_line('Order Number ' || to_char(OrderNumber,'999999')
      ||' isn''t cancelled before ' || TO_CHAR(Cancelldate, 'yyyy/mm/dd'));
    end if;

   end if;
end isCanncelled;
```

בגדול הפונקציה מייצרת טבלה עם כל מספר הזמנה ותאריך ביטול ואם המספר הזמנה זהה לארגומנט והתאריך ביטול לפני או שווה לארגומנט היא תדפיס את ההזמנה, אחרת אם בסוף ההדפסה לא הודפס אף ערך היא תדפיס שההזמנה לא התבטלה לפני התאריך.

דוגמת הרצה:

```
begin
 isCanncelled(3,to_date('01/05/2022','dd/mm/yyyy'));
end;
```


אחיה דנציגר :214092033
אלעד פישר:213924624

הפלט יהיה במקרה הזה

```
Order       3 cancelled
```

במקרה שההזמנה באמת לא בוטלה יהיה מודפס

```
Order Number       2 isn't cancelled before 2022/05/01
```

פרוצדורה 2: בדיקת השכרות עם תוספות

הפרוצדורה הזאת מחפשת את כל ההשכרות שהן עם מספר מסוים של ימים ועם התוספת הספציפית שבתור הארגומנט. למעשה זה יעיל בעיקר כדי לראות את היעילות של הרווחים שלנו כי התוספות מתומחרות פר הזמנה ולא פר יום ולכן זה חשוב לרווחיות שלנו במיוחד.

```
create or replace procedure
ordersWithSpecificExtraAndDays( days integer, extrasInt integer)
is
x number:=0;

cursor extrasPrint is
       select o.order_id as orderID
       from extrasfororders e, orders o
       where e.extra_id = extrasInt and
       o.order_id = e.order_id and o.end_date-o.start_date = days
       order by o.order_id;
sl extrasPrint%Rowtype;
begin
  open extrasPrint;
  if(extrasPrint%Notfound) then
  dbms_output.put_line('No');
  else
    loop
      fetch extrasPrint into sl;
      exit when extrasPrint%notfound;
      dbms_output.put_line(to_char(sl.orderID,'999999'));
    end loop;
    end if;
end ordersWithSpecificExtraAndDays;
```

בגדול הפרוצדורה פשוט תייצר את הטבלה עם כל ההזמנות הרלוונטיות ותדפיס את השורות, אם לא יהיה אף הזמנה כזאת יודפס הודעה מיוחדת

את הפונקציה נריץ ככה:

```
begin
  ordersWithSpecificExtraAndDays(3,1);
end;
```

שזה בקשה להדפיס את כל ההזמנות עם התוספת מספר 1 והזמנה של יום אחד.

התוצאה תהיה:

153
206
340
347
399
417
484
954
965
994
1007
1011
1474
1482
1497
1509
1670
1739
1866
2063
2256
2293
2352
2370
2678
2905
2921
2937
3301
3308
3450
3478
3484
3550
3922
4028
4034
4085
4219
4300

זו רשימה חלקית של ההזמנות הרלוונטיות.