

בסיסי נתונים מסכמת עבודה

מתן לבנטר -208447029

שאלה 1:

רשות הספינות העולמית החליטה להקים מערכת מידע שתנהל את הנתונים על ההפלגות בין השנים 2010 עד 2021.

המערכת תשמור את הנתונים על הספינות, כשלכל ספינה יהיה מזהה ספינה ייחודי וחייב להיות לה גם שם. בנוסף, לכל ספינה יהיו מאפיינים: רוחב אורך ותפוסה וכאשר ספינה תימחק או תתעדכן מהמערכת כל מאפייניה ימחקו. כל ספינה תהיה שייכת לחברה אחת בלבד. לחברה יהיה מזהה חברה, יחוייב להיות לה שם חברה ושם המדינה בה החברה ממוקמת. כמו כן, אם חברה תימחק מהמערכת כל הספינות שהיו שייכות לה יימחקו מהמערכת.

יתר על כן, בכל הפלגה יהיה מזהה נמל שממנו הספינה תצא וגם יהיה מזהה נמל אליה הספינה תגיע, יישמר גם תאריך ההפלגה ומזהה הספינה שבאמצעותה מפליגים. כמו כן, המערכת תשמור נתונים של נמלים, כשלכל נמל יהיה מזהה נמל וחייב יהיה להיות גם שם הנמל ושם המדינה בה הנמל שוכן.

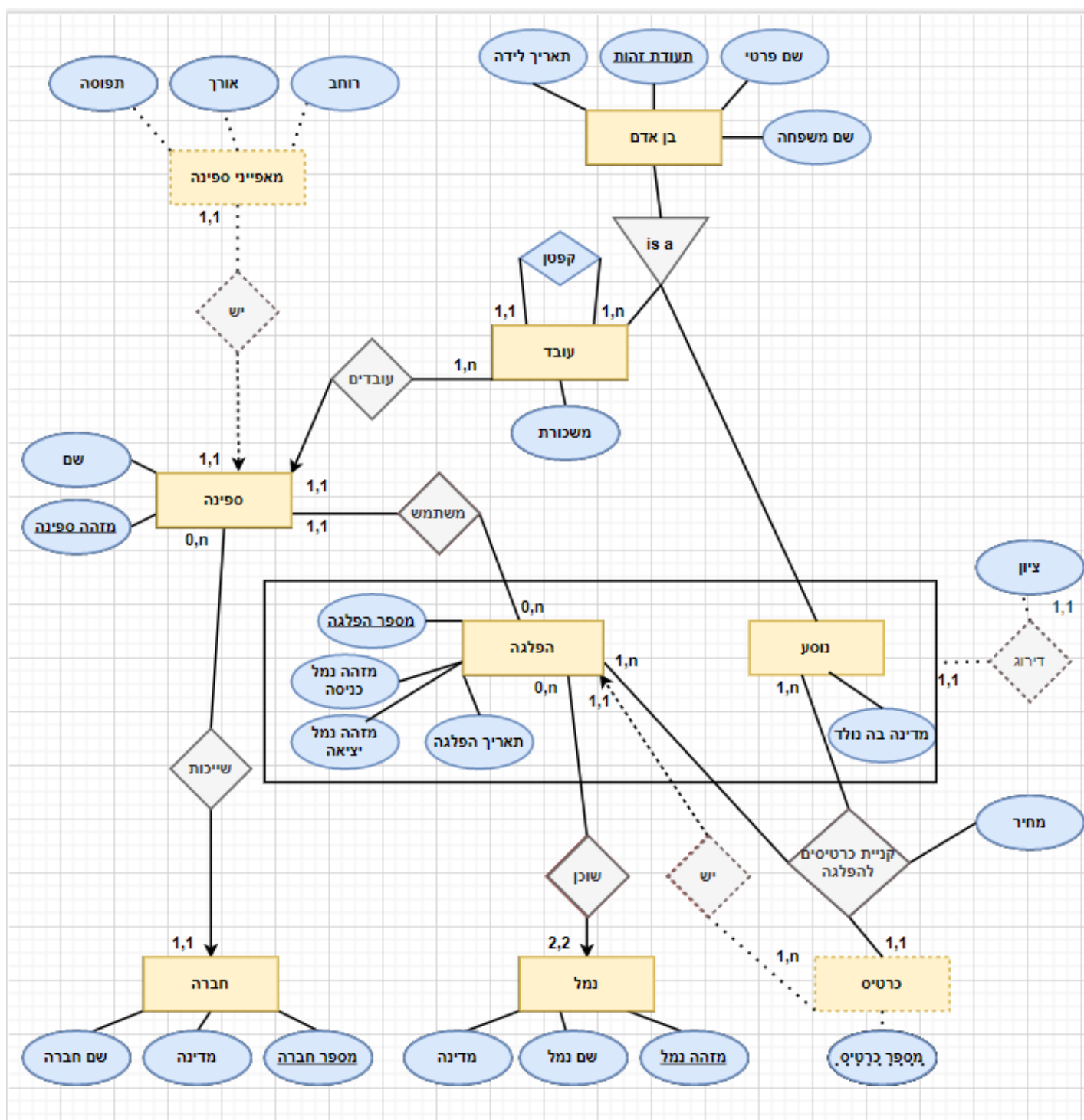
יש להדגיש כי לא ייתאפשר שנמל היציאה ונמל היעד יהיה זהה.

כמו כן, המערכת תשמור את כל האנשים שיכולים להיות נוסעים או עובדים על הספינה, לכל בן אדם יהיה מספר תעודת זהות, יחוייב להיות לבן האדם שם פרטי ושם משפחה וגם יישמר תאריך הלידה שלו. לכל נוסע יישמר גם שם המדינה בה נולד. לכל עובד יהיה שמור גם שכרו ותעודת הזהות של העובד האחראי עליו. יש להדגיש כי כל עובד יוכל לעבוד רק על ספינה אחת.

כאשר תתקיים הפלגה יוקצו לה כרטיסים, ולכל כרטיס יהיה מזהה כרטיס ייחודי, ויישמרו גם כל מחירי הכרטיסים שאותם נוסעים שילמו.

עבור כל הפלגה יחוייב הנוסע לתת דירוג על איכות ההפלגה מספר בין 1-5 כאשר 5 הציון הגבוה ביותר ו-1 הציון הנמוך ביותר.

שאלה 2:



ספינה :

Attribute name	Attribute description	Example value
Ship_ID	A 4-digits number that uniquely describes the ship. Between(1000-1499)	1111
Name_ship	Name given to the ship.	'ulopez'

מאפייני ספינה :

Attribute name	Attribute description	Example value
Ship_ID	A 4-digits number that uniquely describes the ship. Between(1000-1499)	1111
Ship_width	The width of the ship is at its widest point. Between(50-150)	100 m
Ship_length	The length of the ship is from the frontmost point of the ship to the rearmost point. Between(200-400)	300 m
Ship_occupancy	Ship occupancy is the volume of cargo a ship is capable of carrying. Between(20000-7000)	50000 T

חברה :

Attribute name	Attribute description	Example value
Company_id	A number that uniquely describes the company. Between(1-300)	3
Company_country	Company location.	Isreal
Company_Name	Name given to the company.	'Isreal_Shipping_Company'

נמל :

Attribute name	Attribute description	Example value
Port ID	A 3-digit number that uniquely describes the port. Between(300-400)	333
Port_name	A name given to a port.	'vedwards'
Port_country	Port location.	Isreal

הפלגה :

Attribute name	Attribute description	Example value
Sail_id	A 5-digits number that uniquely describes the sailing.	12345
port_id_enter	A 3-digit number that uniquely describes the port. Between(300-400) End point of the sail	333
port_id_exit	A 3-digit number that uniquely describes the port. Between(300-400) start point of the sail	353
Sail_date	Date which occurs sailing. From 2010	10/10/2010

נוסע :

Attribute name	Attribute description	Example value
passenger_ID	A 8-9-digits number that uniquely describes the passenger.	123456789
First name	First name of passenger	Matan
Last Name	last name of passenger	Leventer
Date of birth	Date the passenger was born.	21/10/1996
Country where he was born	Country where he was born.	Isreal

עובד :

Attribute name	Attribute description	Example value
Employee_id	A 8-9-digits number that uniquely describes the employee.	123456789
First name	First name of employee	Matan
Last Name	last name of employee	Leventer
Date of birth	Date the employee was born.	21/10/1996
Employee_id_captain	A 8-9-digits number that uniquely describes the captain of the employee	987654321
Employee salary	The amount of salary that each employee on the ship earns per sailing.	5000

כרטיסים :

Attribute name	Attribute description	Example value
Sail_id	A 5-digits number that uniquely describes the sailing.	12345
Ticket Number	A 8-9 digits number that uniquely describes the card.	992676583

קניית כרטיסים להפלגה :

Attribute name	Attribute description	Example value
passenger_ID	A 8-9-digits number that uniquely describes the passenger.	123456789
Sail_id	A 5-digits number that uniquely describes the sailing.	12345
Ticket Number	A 8-9 digits number that uniquely describes the card.	992676583
Ticket_price	Price given to the ticket according to the type of ticket he bought. Between(1000-7500)	1500

דירוג :

Attribute name	Attribute description	Example value
passenger_ID	A 8-9-digits number that uniquely describes the passenger.	123456789
Sail_id	A 5-digits number that uniquely describes the sailing.	12345
Sail_rating	Each passenger on the ship can give a rating according to the feelings and experience and how much he enjoyed being on the cruise(number between 1-5).	5

שאלה 3:

הרלציות המתקבלות:

- ספינה (מזהה ספינה, שם, מספר חברה)
- חברה (מספר חברה, מדינה, שם חברה)
- נמל (מזהה נמל, מדינה, שם נמל)
- הפלגה (מספר הפלגה, מזהה נמל כניסה, מזהה נמל יציאה, תאריך הפלגה, מזהה ספינה)
- נוסע (תעודת זהות, שם פרטי, שם משפחה, תאריך לידה, מדינה בה נולד)
- עובד (תעודת זהות, שם פרטי, שם משפחה, תאריך לידה, משכורת, תעודת זהות של עובד שאחראי עליו, מזהה ספינה)
- מאפייני ספינה (מזהה ספינה, אורך, רוחב, תפוסה)
- כרטיס (מספר הפלגה, מספר כרטיס)
- קניית כרטיסים להפלגה (מספר הפלגה, תעודת זהות של נוסע, מספר כרטיס, מחיר)
- דירוג (מספר הפלגה, תעודת זהות של נוסע, ציון)

ship

<u>Ship id (pk)</u>	Ship_name	Company_id (fk)
---------------------	-----------	-----------------

company

<u>Company id (pk)</u>	Company_name	Company_country
------------------------	--------------	-----------------

port

<u>port id (pk)</u>	port_name	port_country
---------------------	-----------	--------------

sail

<u>Sail id (pk)</u>	Port_id_enter(fk)	Port_id_exit(fk)	Sail_date	Ship_id(fk)
---------------------	-------------------	------------------	-----------	-------------

passenger

<u>Passenger id(pk)</u>	Passenger_firstname	Passenger_lastname	Passenger_birthday	Passenger_countryborn
-------------------------	---------------------	--------------------	--------------------	-----------------------

employee

<u>Employee id(pk)</u>	Employee_firstname	Employee_lastname	Employee_birthday	Employee_salary	Employee_id_captain	Ship_id(fk)
------------------------	--------------------	-------------------	-------------------	-----------------	---------------------	-------------

Ship_features

<u>Ship id(pk+fk)</u>	Ship_width	Ship_length	Ship_occupancy
-----------------------	------------	-------------	----------------

ticket

<u>Sail id(pk +fk)</u>	<u>Ticket id(pk)</u>
------------------------	----------------------

Buying_tickets_sailing

<u>Passenger id(pk+fk)</u>	<u>Sail id(pk+fk)</u>	<u>Ticket id(pk+fk)</u>	Ticket_price
----------------------------	-----------------------	-------------------------	--------------

Sail_rating

<u>Sail id(pk+fk)</u>	<u>Passenger id(pk+fk)</u>	Sail_rating
-----------------------	----------------------------	-------------

שאלה 4:

כל הטבלאות מקיימות את כללי הנרמול לפי BCNF, נדגים זאת על ידי שתי טבלאות:

- הפלגה (מספר הפלגה, מזהה נמל כניסה, מזהה נמל יציאה, תאריך הפלגה, מזהה ספינה)

המפתח היחיד בטבלה הוא מספר הפלגה.

ניתן לראות שכל עמודה אחרת בטבלה תלוייה אך ורק במפתח ולא באף עמודה אחרת.

מספר הפלגה -> מזהה נמל כניסה

מספר הפלגה -> מזהה נמל יציאה

מספר הפלגה -> תאריך הפלגה

מספר הפלגה -> מזהה ספינה

- דירוג (מספר הפלגה, תעודת זהות של נוסע, ציון)

המפתחות היחידים בטבלה הוא השילוב בין מספר הפלגה, תעודת זהות של נוסע.

ניתן לראות שכל עמודה אחרת בטבלה תלוייה אך ורק במפתח ולא באף עמודה אחרת.

מספר הפלגה, תעודת זהות של נוסע -> ציון

בשתי הטבלאות, שום עמודה אינה תלויה בשום דבר אחר מלבד המפתח ולכן הן מקיימות BCNF.

שאלה 5:

קובץ סקריפט.

שאלה 6:

1---

שלוף את כל תעודות הזהות של הנוסעים ששם הפרטי הוא גיימס שנתנו דירוג מעל ל-2 ומחיר הכרטיס ששילמו גדול מ-3000.

2---

הציגו לכל נוסע את תעודת הזהות שם פרטי ושם משפחה ואת כמות ההפלגות שאליהם הפליג.

3---

שלוף את תעודות הזהות של כל הנוסעים שלא דירגו אף הפלגה בדירוג 5.

4---

שלוף את כל המדינות שבהם נולדו מעל ל-90 נוסעים משנת 1980.

5---

שלוף את כל מזההי הספינות שיצאו להפלגה מנמל היציאה כמו הפלגה מספר 23904.

6---

הציגו את תעודות הזהות של כל הנוסעים ומחירי הכרטיסים הנמוכים ביותר בכל הפלגה.

7---

הציגו את תעודות הזהות של כל הנוסעים שדירגו את כל האפשרויות של הציון (1,2,3,4,5).

8---

הציגו את טבלת ההיררכיה של כל העובדים והחזירו את תעודת הזהות, שם פרטי, שם משפחה, תעודת זהות של הקפטן שלהם ואת טבלת ההיררכיה שלהם כמספר דרגות הניהול מעובד ועד הקפטן הראשי של הספינה שאין מעליו אף אחד ודרגתו אחד.

9---

שלוף את כל תאריכים שבאותו היום נולדו בדיוק שלוש הנוסעים ושלוש העובדים.

10---

שלוף את כל תאריכי ההפלגות, שיצאו להפלגה ספינות מחברת הספינות של ישראל ומזהה הספינות שלהם גדול מ1200.

שאלה 7:

הכנסה של נתונים בעזרת Pycharm.

```
import psycopg2
from faker import Faker
import random
import pycountry

def connect_to_db(dbname, user, password):
    con = psycopg2.connect(f"dbname={dbname} user={user} password={password}")
    cursor = con.cursor()
    return con, cursor
```

```
def close_communication(cur, conn):
    cur.close()
    conn.close()
```

```
def get_countries():
    list=[]
    for x in pycountry.countries:
        list.append(x.name)
    return list
```

```
list_country=get_countries() # list of countries
num_rows=10000 # number of rows i want to enter
salary_1=[25000,30000,50000,45000,35000,20000,55000,22500,37500,47500,32500,27500,22150] # salary captain
salary_2=[16000,16500,17000,17500,18000,18500,19000,19500] # salary
salary_3=[13500,12000,11000,10000,10500,15000,13000,12500,11500] # salary
salary_4=[4500,3000,3500,6500,8000,9000,8500,9500,7000,4000,6000,6500,5500,7500,8500] # salary
salary_5=[500,750,1000,1250,1500,1750,2000] # salary
conn,cur =connect_to_db("matan", "postgres", "iNx8yqpa!")
populate_db(cur)
conn.commit()
close_communication(cur,conn)
```

company

```
def populate_db(num_rows, cur):
    e=[]
    for i in range(num_rows):
        id=random.randint(1,300)
        while id in e:
            id = random.randint(1, 300)
        e.append(id)
        country = list_country[id%(len(list_country))]
        name=(country+' shipping company')
        a=(id,name, country)
        cur.execute("INSERT INTO company (company_id,company_name,company_country) values (%s,%s,%s)"%(a))
```

Ship

```
def populate_db(num_rows, cur):
    faker = Faker()
    cur.execute("select company_id from company")
    all = cur.fetchall()
    id_company=[]
    for j in all:
        id_company.append(j[0])
    e=[]
    for i in range(num_rows):
        id=random.randint(1000,1500)
        while id in e:
            id=random.randint(1000,1500)
        e.append(id)
        name1=faker.email()
        name=name1[:name1.index("@")]
        companyid= id_company[id%(len(id_company))]
        a=(id, name, companyid)
        cur.execute("INSERT INTO ship (ship_id,ship_name,company_id) values (%s,%s,%s)" % (a))
```

Ship features

```
def populate_db_ship_features(cur):
    cur.execute("select ship_id from ship")
    all = cur.fetchall()
    for i in all:
        num=i
        width=random.randint(50,150)
        length= random.randint(200,400)
        occ=(width//10*length//10)*100+10000
        a=(num,width,length,occ)
        cur.execute("INSERT INTO ship_features (ship_id,ship_width,ship_length,ship_occupancy) values (%s,%s,%s,%s)" % (a))
```

port

```
def populate_db(num_rows, cur):
    faker = Faker()
    cur.execute("select distinct(company_country) from company")
    all = cur.fetchall()
    country=[]
    for j in all:
        country.append(j[0])
    e=[]
    for i in range(num_rows):
        id=random.randint(300,400)
        while id in e:
            id = random.randint(300, 400)
        e.append(id)
        name1=faker.email()
        name=name1[:name1.index("@")]
        country= country[id%(len(country))]
        a=(id, name, country)
        cur.execute("INSERT INTO port (port_id,port_name,port_country) values (%s,%s,%s)" % (a))
```

Sail

```
def populate_db(num_rows, cur):
    faker = Faker()
    cur.execute("select port_id from port")
    all = cur.fetchall()
    e=[]
    r=[]
    z=[]
    for i in all:
        e.append(i[0])
    cur.execute("select ship_id from ship")
    all = cur.fetchall()
    for i in all:
        z.append(i[0])
    for x in range(num_rows):
        random_num=random.randint(10000,99999)
        random_num_1 = random.randint(10000, 99999)
        while random_num%50 ==random_num_1%50 or random_num in r:
            random_num = random.randint(10000,99999)
            random_num_1 = random.randint(10000, 99999)
        r.append(random_num)
        port_id_enter=e[random_num%50]
        port_id_exist=e[random_num_1 % 50]
        ship_id=z[random_num%200]
        date=faker.date()
        while (date<'2010'):
            date = faker.date()
        a=(random_num, port_id_enter, port_id_exist, date, ship_id)
        cur.execute("INSERT INTO sail (sail_id, port_id_enter, port_id_exit, sail_date, ship_id) values (%s,%s,%s,%s,%s)", (a))
```

Passenger

```
def populate_db(num_rows, cur):
    faker = Faker()
    sum=0
    e=[]
    z=[]
    for j in range(num_rows):
        sum=0
        for i in range(9):
            random_num = random.randint(0,9)
            sum+=random_num*10**i
        while sum in e:
            sum=0
            for i in range(9):
                random_num = random.randint(0, 9)
                sum += random_num * 10 ** i
        e.append(sum)
        name=faker.first_name()
        last=faker.last_name()
        date=faker.date()
        while (date>'2000' or date<'1930'):
            date = faker.date()
        con=list_country[sum%230]
        a=(sum, name, last, date, con)
        cur.execute("INSERT INTO passenger (passenger_id, passenger_firstname, passenger_lastname, passenger_birthday, passenger_countryborn) values (%s,%s,%s,%s,%s)", (a))
```

Sail rating

```
def populate_db(cur):
    cur.execute("select passenger_id, sail_id from buying_tickets_sailing")
    all = cur.fetchall()
    for i in all:
        number = random.randint(1, 5)
        a=(i[0], i[1], number)
        cur.execute("INSERT INTO sail_rating (passenger_id , sail_id, sail_rating) values (%s,%s,%s)", (a))
```

Employee

```
168
169 def populate_db(num_rows, cur):
170     faker=Faker()
171     cur.execute("select ship_id from ship")
172     all = cur.fetchall()
173     e=[]
174     r=[]
175     j=[]
176     m=[]
177     l=[]
178     ship=[]
179     for i in all:
180         ship.append(i[0])
181     for iter in range(num_rows):
182         sum=0
183         for i in range(9):
184             random_num = random.randint(0, 9)
185             sum += random_num * 10 ** i
186         while sum in e:
187             sum = 0
188             for i in range(9):
189                 random_num = random.randint(0, 9)
190                 sum += random_num * 10 ** i
191         if iter<len(ship):
192             caption=sum
193             r.append((sum,ship[iter%200]))
194             name=faker.first_name()
195             last=faker.last_name()
196             date=faker.date()
197             while (date>'1990' or date<'1960'):
198                 date = faker.date()
199             salary=salary_1[sum%len(salary_1)]
200             a=(sum,name,last,date,saly,caption,ship[iter%200])
201             cur.execute("INSERT INTO employee(employee_id,employee_firstname,employee_lastname,employee_birthday,employee_salary,employee_id_captain,ship_id) values (%s,%s,%s,%s,%s,%s,%s)",(a))
202         elif iter<800:
203             caption=r[iter%200][0]
204             j.append((sum,ship[iter%200]))
205             name = faker.first_name()
206             last = faker.last_name()
207             date = faker.date()
208             saly = salary_2[sum % len(salary_2)]
209             while (date > '1990' or date < '1960'):
210                 date = faker.date()
211             a = (sum, name, last, date, saly, caption, ship[iter % 200])
212             cur.execute("INSERT INTO employee(employee_id,employee_firstname,employee_lastname,employee_birthday,employee_salary,employee_id_captain,ship_id) values (%s,%s,%s,%s,%s,%s,%s)",(a))
213         elif iter<2400:
214             caption=j[iter%600][0]
215             m.append((sum,ship[iter%200]))
216             name = faker.first_name()
217             last = faker.last_name()
218             date = faker.date()
219             while (date > '1990' or date < '1960'):
220                 date = faker.date()
221             saly = salary_3[sum % len(salary_3)]
222             a = (sum, name, last, date, saly, caption, ship[iter % 200])
223             cur.execute("INSERT INTO employee(employee_id,employee_firstname,employee_lastname,employee_birthday,employee_salary,employee_id_captain,ship_id) values (%s,%s,%s,%s,%s,%s,%s)",(a))
224         elif iter<9600:
225             caption=m[iter%1600][0]
226             l.append((sum,ship[iter%200]))
227             name = faker.first_name()
228             last = faker.last_name()
229             date = faker.date()
230             while (date > '1990' or date < '1960'):
231                 date = faker.date()
232             saly = salary_4[sum % len(salary_4)]
233             a = (sum, name, last, date, saly, caption, ship[iter % 200])
234             cur.execute("INSERT INTO employee(employee_id,employee_firstname,employee_lastname,employee_birthday,employee_salary,employee_id_captain,ship_id) values (%s,%s,%s,%s,%s,%s,%s)",(a))
235         else:
236             caption = l[iter % 200][0]
237             name = faker.first_name()
238             last = faker.last_name()
239             date = faker.date()
240             while (date > '1990' or date < '1960'):
241                 date = faker.date()
242             saly = salary_5[sum % len(salary_5)]
243             a = (sum, name, last, date, saly, caption, ship[iter % 200])
244             cur.execute("INSERT INTO employee(employee_id,employee_firstname,employee_lastname,employee_birthday,employee_salary,employee_id_captain,ship_id) values (%s,%s,%s,%s,%s,%s,%s)",(a))
245
```

Ticket

```
def populate_db(cur):
    z=[]
    cur.execute("select sail_id from sail")
    all = cur.fetchall()
    for i in all:
        z.append(i[0])
    for r in z:
        e=[]
        for j in range(100):
            sum=0
            for i in range(9):
                random_num = random.randint(0,9)
                sum+=random_num*10**i
            if sum in e:
                print('yes')
                while sum in e:
                    sum=0
                    for i in range(8):
                        random_num = random.randint(0, 9)
                        sum += random_num * 10 ** i
            e.append(sum)
        a=(r,sum)
        cur.execute("INSERT INTO ticket (sail_id,ticket_id) values (%s,%s)",(a))
```

Buying tickets sailing

```
def populate_db(cur):
    passenger=[]
    ticket_sail=[]
    e=[]
    cur.execute("select passenger_id from passenger")
    all = cur.fetchall()
    for i in all:
        passenger.append(i[0])
    cur.execute("select sail_id,ticket_id from ticket")
    all = cur.fetchall()
    for i in all:
        ticket_sail.append((i[0],i[1]))
    for x in range(len(passenger)):
        list1 = []
        if (passenger[x]%3==0):
            number=1
        elif (passenger[x]%5==0):
            number=2
        elif (passenger[x]%8==0):
            number=3
        else:
            number=random.randint(1, 9)
        for j in range(number):
            y=(random.randint(0, 99999))
            while ticket_sail[y][1] in e or ticket_sail[y][0] in list1:
                y = (random.randint(0, 99999))
            list1.append(ticket_sail[y][0])
            e.append(ticket_sail[y][1])
            if (passenger[x]%1799==0):
                u=0
            price = (random.randint(1000, 7500))
            a=(passenger[x],ticket_sail[y][0],ticket_sail[y][1],price)
            cur.execute("INSERT INTO buying_tickets_sailing (passenger_id sail_id ticket_id ticket price) values (%s,%s,%s,%s)",(a))
```

שאלה 8:

שאלתה 1:

```
1 ---1
2 ---3000
3 select distinct(p.passenger_id)
4 from passenger as p inner join sail_rating as sr
5 on p.passenger_id=sr.passenger_id
6 inner join buying_tickets_sailing as bts
7 on bts.passenger_id=p.passenger_id
8 where p.passenger_firstname='James' and sr.sail_rating >2 and bts.ticket_price>3000
9
```

Explain Notifications Messages Data Output Scratch Pad

	passenger_id [PK] integer
1	368465703
2	558304639
3	953439029
4	47542358
5	921091930
6	561546419
7	796468217
8	172091746
9	450090361
10	597970547
11	783426647
12	135707250
13	593292526
14	440349662
15	324790982
16	414647246

✓ Successfully run. Total query runtime: 151 msec. 258 rows affected.

שאלתה 2:

```
10 ---2
11 ---יצא
12 select p.passenger_id,p.passenger_firstname,p.passenger_lastname,count(ticket_id)
13 from passenger as p left outer join buying_tickets_sailing
14 on(p.passenger_id = buying_tickets_sailing.passenger_id)
15 group by p.passenger_id
16
17 ---3
```

Explain Notifications Messages Data Output Scratch Pad

	passenger_id [PK] integer	passenger_firstname character varying (50)	passenger_lastname character varying (50)	count bigint
1	49826217	Jamie	Hernandez	1
2	279292458	Tracy	Palmer	1
3	330497063	Kimberly	Morrison	7
4	602785132	Jeff	Lowery	3
5	760048313	Robert	Stewart	6
6	560389924	Ashley	Christian	1
7	328653342	Kelly	Sexton	1
8	842005971	Eric	Hester	1
9	819728396	David	Miller	4
10	656810610	Latoya	Morris	1
11	462008453	Mercedes	Long	6
12	184366549	John	Anderson	7
13	727214056	Madeline	Parsons	3
14	321414540	Diane	Watson	1
15	247808746	Julia	Erickson	2
16	615136059	Albert	Walker	1

✓ Successfully run. Total query runtime: 290 msec. 25000 rows affected.

שאלתה 3:

```
17 ---3
18 ---5
19 select passenger_id
20 from passenger
21 EXCEPT
22 select passenger_id
23 from sail_rating
24 where sail_rating=5
25
```

שלוף את תעודות הזהות של כל הנוסעים שלא דירגו אף הפלגה בדירוג 5

Explain Notifications Messages **Data Output** Scratch Pad

	passenger_id integer	
1	279292458	
2	49826217	
3	602785132	
4	560389924	
5	328653342	
6	656810610	
7	247808746	
8	201674714	
9	734465856	
10	174374493	
11	123267063	
12	230831656	

✓ Successfully run. Total query runtime: 161 msec. 14042 rows affected.

שאלתה 4:

```
26 ---4
27 ---1980
28 select passenger_countryborn
29 from passenger
30 where passenger_birthday >= '1-1-1980'
31 group by passenger_countryborn
32 having count(*)>90
33
```

שלוף את כל המדינות שבהם נולדו מעל ל-90 נוסעים משנת 1980

Explain Notifications Messages **Data Output** Scratch Pad

	passenger_countryborn character varying (50)	
1	Albania	
2	Malaysia	
3	Qatar	
4	Kazakhstan	
5	Tunisia	
6	Hong Kong	
7	Korea, Democratic People's ...	

✓ Successfully run. Total query runtime: 178 msec. 7 rows affected.

שאלתה 5:

```
34 ---5
35 ---23904 מספר הפלגה מנמל היציאה כמו הפלגה מספר
36 select distinct(ship_id)
37 from sail
38 where port_id_exit=
39 (select port_id_exit
40 from sail
41 where sail_id=23904
42 )
```

Explain Notifications Messages Data Output Scratch Pad

	ship_id	integer
1	1386	
2	1330	
3	1043	
4	1345	
5	1277	
6	1457	
7	1075	
8	1198	
9	1074	
10	1373	
11	1276	
12	1338	
13	1170	
14	1126	
15	1374	

✓ Successfully run. Total query runtime: 101 msec. 20 rows affected.

שאלתה 6:

```
44 ---6
45 ---הציגו את תעודות הזרות של כל הנוסעים ומחירי הכרטיסים הנמוכים ביותר בכל הפלגה-
46 select passenger_id,ticket_price
47 from buying_tickets_sailing as bts
48 where bts.ticket_price =
49 (select min(bts1.ticket_price)
50 from buying_tickets_sailing as bts1
51 where bts.sail_id=bts1.sail_id)
52
53 ---7
54 (1,2,3,4,5) ---הציגו את תעודות הזרות של כל הנוסעים ומחירי הכרטיסים הנמוכים ביותר בכל הפלגה-
```

Explain Notifications Messages Data Output Scratch Pad

	passenger_id	integer	ticket_price	integer
1	277395263		1101	
2	819498922		1004	
3	84830783		1031	
4	881718435		1042	
5	215318870		1213	
6	761791900		1037	
7	8520725		1178	
8	846304679		1077	
9	315134849		1094	
10	542033249		1014	
11	554449999		1017	
12	18802006		1008	
13	354522740		1007	
14	64259452		1171	
15	404550145		1067	

✓ Successfully run. Total query runtime: 12 secs 345 msec. 1007 rows affected.

שאלתה 7:

```
53 ---7
54 --- (1,2,3,4,5) הציון של האפשרויות של כל הנוסעים שדירגו את כל תעודות הזהות של כל
55 select distinct(passenger_id) from
56 (
57 select p.passenger_id
58 from passenger as p join sail_rating as sr
59 on p.passenger_id = sr.passenger_id
60 ) as sr
61 where not exists
62 (
63 select (sail_rating)
64 from sail_rating as sr1
65 where not exists
66 (
67 select (sail_rating)
68 from sail_rating as sr2
69 where sr2.passenger_id=sr.passenger_id and sr1.sail_rating=sr2.sail_rating
70 )
71 )
```

Explain Notifications Messages Data Output Scratch Pad

	passenger_id [PK] integer
1	616001
2	3353446
3	3619186
4	3860606
5	3879922
6	4558577
7	4683514

✓ Successfully run. Total query runtime: 1 min 45 secs. 1382 rows affected.

שאלתה 8:

```
73 ---8
74 --- הציון של כל תעודות הזהות של כל הנוסעים שדירגו את כל תעודות הזהות של כל הנוסעים שדירגו את כל תעודות הזהות של כל
75 ---1 הציון של כל תעודות הזהות של כל הנוסעים שדירגו את כל תעודות הזהות של כל הנוסעים שדירגו את כל תעודות הזהות של כל
76
77 with recursive emtable as
78 (
79 select captain.employee_id,captain.employee_firstname,captain.employee_lastname,captain.employee_id_captain,1 as emlevel
80 from employee as captain
81 where captain.employee_id_captain=captain.employee_id
82 union all
83 select emp.employee_id,emp.employee_firstname,emp.employee_lastname,emp.employee_id_captain,emtable.emlevel +1
84 from employee as emp
85 join emtable
86 on emp.employee_id_captain = emtable.employee_id
87 where emp.employee_id_captain!=emp.employee_id
88 )
89 select *
90 from emtable
```

Explain Notifications Messages Data Output Scratch Pad

	employee_id integer	employee_firstname character varying (50)	employee_lastname character varying (50)	employee_id_captain integer	emlevel integer
1	714492831	Malik	Schwartz	714492831	1
2	348443420	Kyle	Smith	348443420	1
3	459221805	Shelly	Peterson	459221805	1
4	495310332	Stephen	Montoya	495310332	1
5	8753594	Brent	Thomas	8753594	1
6	460651920	Sally	Davis	460651920	1
7	676464402	Marcus	Shaw	676464402	1

✓ Successfully run. Total query runtime: 176 msec. 10000 rows affected.

שאלתה 9:

```
91
92 ---9
93 ---שלוף את כל תאריכים שבאותו היום נולדו בדיוק שלוש תנוסעים ושלוש הטובדים
94 select passenger_birthday
95 from passenger
96 group by passenger_birthday
97 having count(*)=3
98 intersect
99 select employee_birthday
100 from employee
101 group by employee_birthday
102 having count(*)=3
103
104
```

Explain Notifications Messages Data Output Scratch Pad

	passenger_birthday date	
1	1979-09-08	
2	1974-06-20	
3	1975-07-30	
4	1987-09-06	
5	1982-03-21	
6	1988-08-30	
7	1982-07-11	
8	1989-09-26	
9	1973-06-21	
10	1975-10-23	
11	1979-10-01	
12	1983-10-30	
13	1974-08-17	

✓ Successfully run. Total query runtime: 169 msec. 159 rows affected.

שאלתה 10:

```
104
105 ---10
106 ---שלוף את כל תאריכי ההפלגות, שיצאו להפלגה ספינות מחברת הספינות של ישראל ומזהה הספינות שלהם גדול מ1200
107 select sa.sail_date
108 from ship as s join company as c
109 on s.company_id=c.company_id
110 join sail as sa
111 on s.ship_id=sa.ship_id
112 where c.company_country ='Israel' and s.ship_id >1200
113
114
115
```

Explain Notifications Messages Data Output Scratch Pad

	sail_date date	
1	2018-10-24	
2	2018-05-26	
3	2016-11-20	
4	2018-05-15	
5	2012-09-23	
6	2020-12-26	
7	2015-04-29	
8	2020-08-29	
9	2017-08-23	
10	2016-06-06	
11	2014-08-29	
12	2012-12-04	
13	2020-09-03	

✓ Successfully run. Total query runtime: 110 msec. 28 rows affected.

שאלה 9:

1. שלוף את כל תעודות הזהות של הנוסעים ששם הפרטי הוא גיימס שנתנו דירוג מעל ל-2 ומחיר הכרטיס ששילמו גדול מ-3000.

- $\text{Passenger_james} \leftarrow \pi_{\text{passenger_id}} \sigma_{(\text{passenger_firstname}='James')} (\text{Passenger})$
- $\text{Sail_rating_over_2} \leftarrow \pi_{\text{passenger_id}} \sigma_{(\text{sail_rating} > 2)} (\text{Sail_rating})$
- $\text{buying_tickets_sailing_over_3000} \leftarrow \pi_{\text{passenger_id}} \sigma_{(\text{ticket_price} > 3000)} (\text{buying_tickets_sailing})$
- $\text{Passenger_james_Sail_rating_over_2} \leftarrow \pi_{\text{passenger_id}} (\text{Passenger_james} \bowtie_{(\text{passenger_id} = \text{passenger_id})} \text{Sail_rating_over_2})$
- $\text{Passenger_james_Sail_rating_over_2_buying_tickets_sailing_over_3000} \leftarrow \pi_{\text{passenger_id}} (\text{Passenger_james_Sail_rating_over_2} \bowtie_{(\text{passenger_id} = \text{passenger_id})} \text{buying_tickets_sailing_over_3000})$

2. הציגו לכל נוסע את תעודת הזהות שם פרטי ושם משפחה ואת כמות ההפלגות שאליהם יצא.

- $\text{New_passenger} \leftarrow \pi_{\text{passenger_id}, \text{passenger_firstname}, \text{passenger_lastname}} (\text{passenger})$
- $\text{Passenger_buying_tickets_sailing} \leftarrow \pi_{\text{passenger_id}, \text{passenger_firstname}, \text{passenger_lastname}, \text{ticket_id}} (\text{New_passenger} \bowtie_{(\text{passenger_id} = \text{passenger_id})} \text{buying_tickets_sailing})$
- $\text{Result} (\text{passenger_id}, \text{number_of_sails}, \text{passenger_firstname}, \text{passenger_lastname}) \leftarrow \text{passenger_id} \bowtie \text{COUNT ticket_id}, \text{passenger_firstname}, \text{passenger_lastname} (\text{Passenger_buying_tickets_sailing})$

3. שלוף את תעודות הזהות של כל הנוסעים שלא דירגו אף הפלגה בדירוג 5.

- $\text{all_passenger} \leftarrow \pi_{\text{passenger_id}} (\text{passenger})$
- $\text{passenger_rating_5} \leftarrow \pi_{\text{passenger_id}} \sigma_{(\text{sail_rating}=5)} (\text{sail_rating})$
- $\text{passenger_without_rating_5} \leftarrow \text{all_passenger} - \text{passenger_rating_5}$

4. שלוף את כל המדינות שבהם נולדו מעל ל-90 נוסעים משנת 1980.

- $\text{Passenger_after_1980} \leftarrow \pi_{\text{passenger_id}, \text{passenger_countryborn}} \sigma_{(\text{passenger_countryborn} > '1-1-1980')} (\text{passenger})$
- $\text{Passenger_after_1980_1} (\text{passenger_countryborn}, \text{number_of_passenger_born}) \leftarrow \text{passenger_countryborn} \bowtie \text{COUNT passenger_id} (\text{Passenger_after_1980})$
- $\text{Result} \leftarrow \pi_{\text{passenger_countryborn}} \sigma_{(\text{number_of_passenger_born} > 90)} (\text{Passenger_after_1980_1})$

5. שלוף את כל מזההי הספינות שיצאו להפלגה מנמל היציאה כמו הפלגה מספר 23904.

- $Sail_23904 \leftarrow \pi_{port_id_exit} (\sigma_{(sail_id=23904)} (sail))$
- $result \leftarrow \pi_{ship_id} (Sail_23904 \bowtie_{(port_id_exit = port_id_exit)} sail)$

6. הציגו את תעודות הזהות של כל הנוסעים ומחירי הכרטיסים הנמוכים ביותר בכל הפלגה.

- $Result (passenger_id, ticket_price) \leftarrow sail_id \Join \min_{ticket_price} (buying_tickets_sailing)$

7. הציגו את תעודות הזהות של כל הנוסעים שדירגו את כל האפשרויות של הצינון (1,2,3,4,5).

- $Options_rate \leftarrow \pi_{sail_rating} (sail_rating)$
- $All_paseenger_sail_rating \leftarrow \pi_{passenger_id, sail_rating} (passenger \bowtie_{(passenger_id = passenger_id)} sail_rating)$
- $Result \leftarrow \pi_{passenger_id} (All_paseenger_sail_rating \div Options_rate)$

8. הציגו את טבלת ההיררכיה של כל העובדים והחזירו את תעודת הזהות, שם פרטי, שם משפחה, תעודת זהות של הקפטן שלהם וטבלת ההיררכיה שלהם כמספר דרגות הניהול מעובד ועד הקפטן הראשי של הספינה שאין מעליו אף אחד ודרגתו אחד.

- $Employee_lev1 \leftarrow \pi_{employee_id, employee_firstname, employee_lastname, employee_id_captain, 1 \text{ as } emlevel} (\sigma_{(employee_id_captain = employee_id)} (employee))$
- $Employee_lev2 \leftarrow \pi_{employee_id, employee_firstname, employee_lastname, employee_id_captain, emlevel + 1} (\sigma_{(employee_lev1.employee_id = employee.employee_id_captain)} (employee))$
- $Employee_lev3 \leftarrow \pi_{employee_id, employee_firstname, employee_lastname, employee_id_captain, emlevel + 1} (\sigma_{(employee_lev2.employee_id = employee.employee_id_captain)} (employee))$
- $Employee_lev4 \leftarrow \pi_{employee_id, employee_firstname, employee_lastname, employee_id_captain, emlevel + 1} (\sigma_{(employee_lev3.employee_id = employee.employee_id_captain)} (employee))$
- $Employee_lev5 \leftarrow \pi_{employee_id, employee_firstname, employee_lastname, employee_id_captain, emlevel + 1} (\sigma_{(employee_lev4.employee_id = employee.employee_id_captain)} (employee))$

..... employee על הפעולה כל עוד יש רשומות

- $result \leftarrow Employee_lev1 \cup Employee_lev2 \cup Employee_lev3 \cup Employee_lev4 \cup Employee_lev5 \dots \cup Employee_levn$

9. שלוף את כל תאריכים שבאותו היום נולדו בדיוק שלוש הנוסעים ושלוש העובדים.

- $\text{Passenger_date}(\text{passenger_birthday}, \text{number_of_passenger_born}) \leftarrow \text{passenger_birthday} \bowtie \text{COUNT}_{\text{passenger_id}}(\text{passenger})$
- $\text{Passenger_date_3} \leftarrow \pi_{\text{passenger_birthday}}(\sigma_{(\text{number_of_passenger_born}=3)}(\text{Passenger_date}))$
- $\text{employee_date}(\text{employee_birthday}, \text{number_of_employee_born}) \leftarrow \text{employee_birthday} \bowtie \text{COUNT}_{\text{employee_id}}(\text{employee})$
- $\text{employee_date_3} \leftarrow \pi_{\text{employee_birthday}}(\sigma_{(\text{number_of_employee_born}=3)}(\text{employee_date}))$
- $\text{result} \leftarrow \text{employee_date_3} \cap \text{Passenger_date_3}$

10. שלוף את כל תאריכי ההפלגות, שיצאו להפלגה ספינות מחברת הספינות של ישראל ומזהה הספינות שלהם גדול מ1200.

- $\text{Company_Isreal} \leftarrow \pi_{\text{company_id}}(\sigma_{(\text{company_country}='Israel')}(\text{Company}))$
- $\text{Ship_id_over_1200} \leftarrow \pi_{\text{ship_id}, \text{company_id}}(\sigma_{(\text{ship_id}>1200)}(\text{Ship}))$
- $\text{Sail_date} \leftarrow \pi_{\text{ship_id}, \text{sail_date}}(\text{Sail_date})$
- $\text{Company_Isreal_Ship_id_over_1200} \leftarrow \pi_{\text{ship_id}}(\text{Ship_id_over_1200} \bowtie_{(\text{company_id} = \text{company_id})} \text{Company_Isreal})$
- $\text{Company_Isreal_Ship_id_over_1200_Sail_date} \leftarrow \pi_{\text{sail_date}}(\text{Sail_date} \bowtie_{(\text{ship_id} = \text{ship_id})} \text{Company_Isreal_Ship_id_over_1200})$

שאלה 10:

1. שלוף את כל תעודות הזהות של הנוסעים ששם הפרטי הוא גיימס שנתנו דירוג מעל ל-2 ומחיר הכרטיס ששילמו גדול מ-3000.

{t. passenger_id | passenger (t) AND passenger_firstname='James' AND $\forall d$ (sail_rating (d) AND d. sail_rating >2 AND t. passenger_id =d.passenger_id) AND $\forall e$ (buying_tickets_sailing (e) AND e. ticket_price>3000 AND e. passenger_id =t. passenger_id)}

2. שלוף את תעודות הזהות של כל הנוסעים שלא דירגו אף הפלגה בדירוג 5.

{t.passenger_id|passenger (t) AND $\forall d$ (sail_rating(d) AND NOT d.sail_rating=5 AND t.passenger_id =d. passenger_id)}

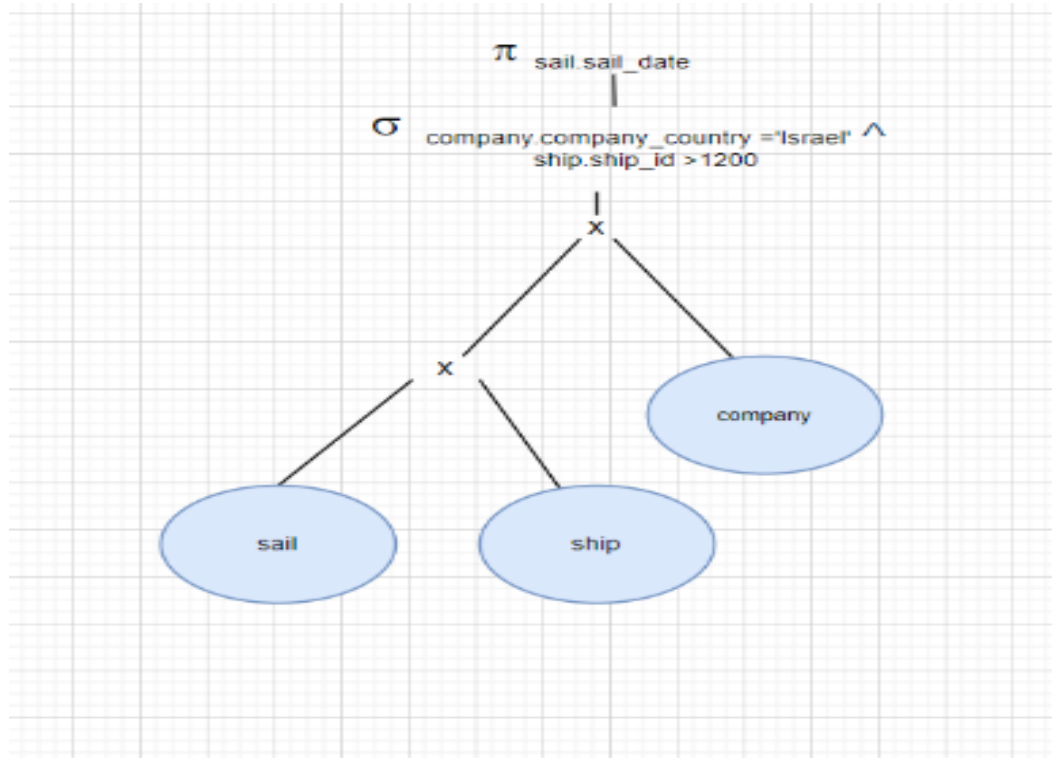
3. שלוף את כל מזההי הספינות שיצאו להפלגה מנמל היציאה כמו הפלגה מספר 23904.

{t.ship_id| sail(t) AND $\forall d$ (sail (d) AND d.sail_id=23904 AND t. port_id_exit =d. port_id_exit)}

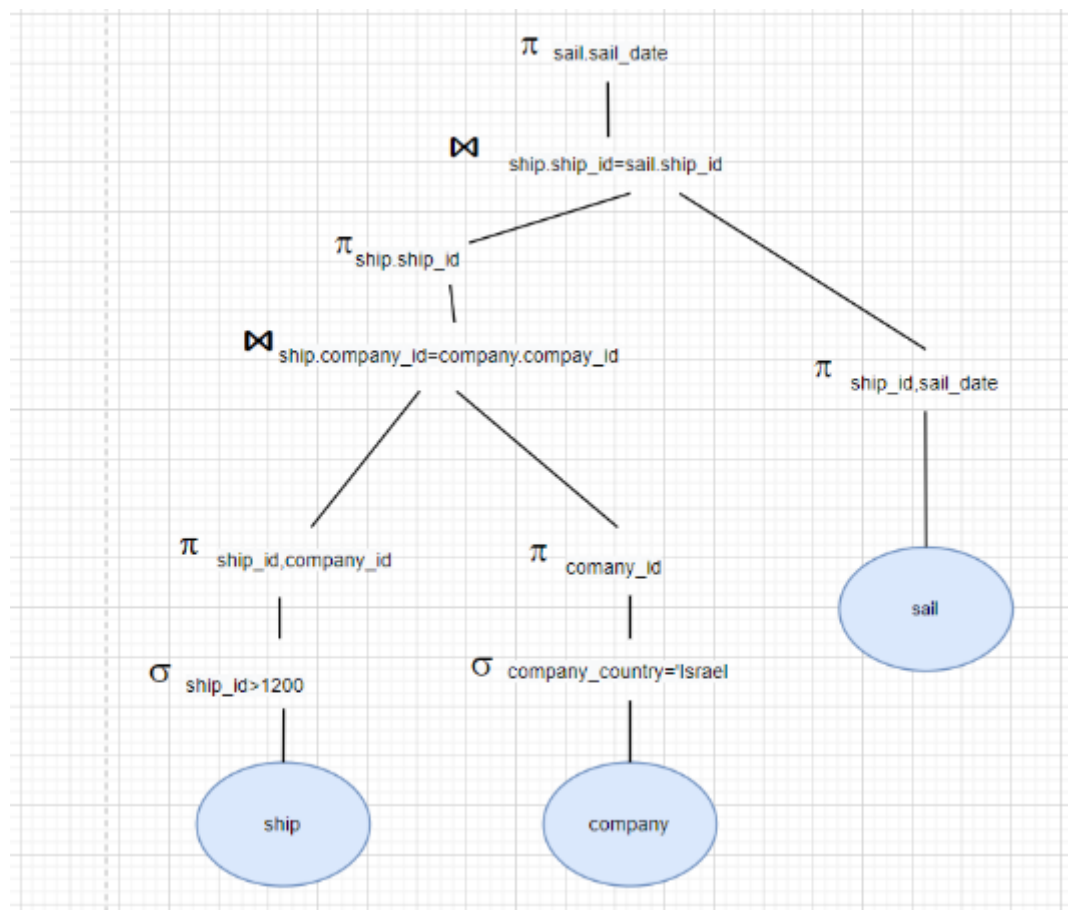
שאלה 11:

1. שלוף את כל תאריכי ההפלגות, שיצאו להפלגה ספינות מחברת הספינות של ישראל ומזהה הספינות שלהם גדול מ1200.

עץ שאילתה ראשוני

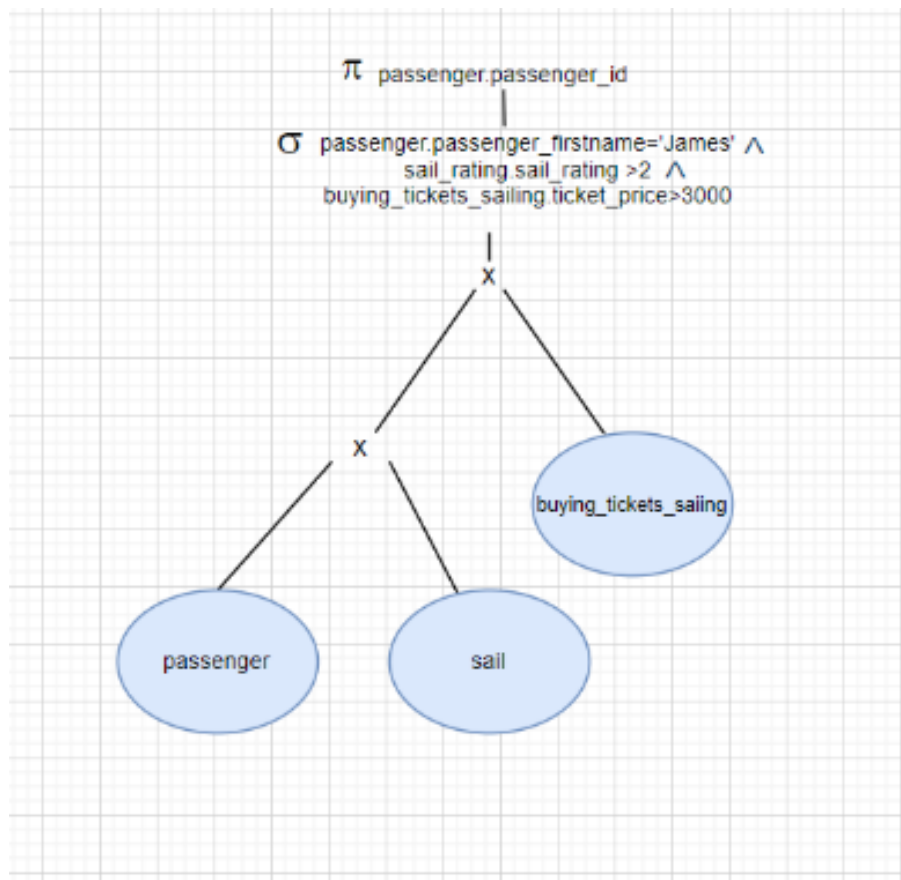


עץ שאילתה אופטימאלי

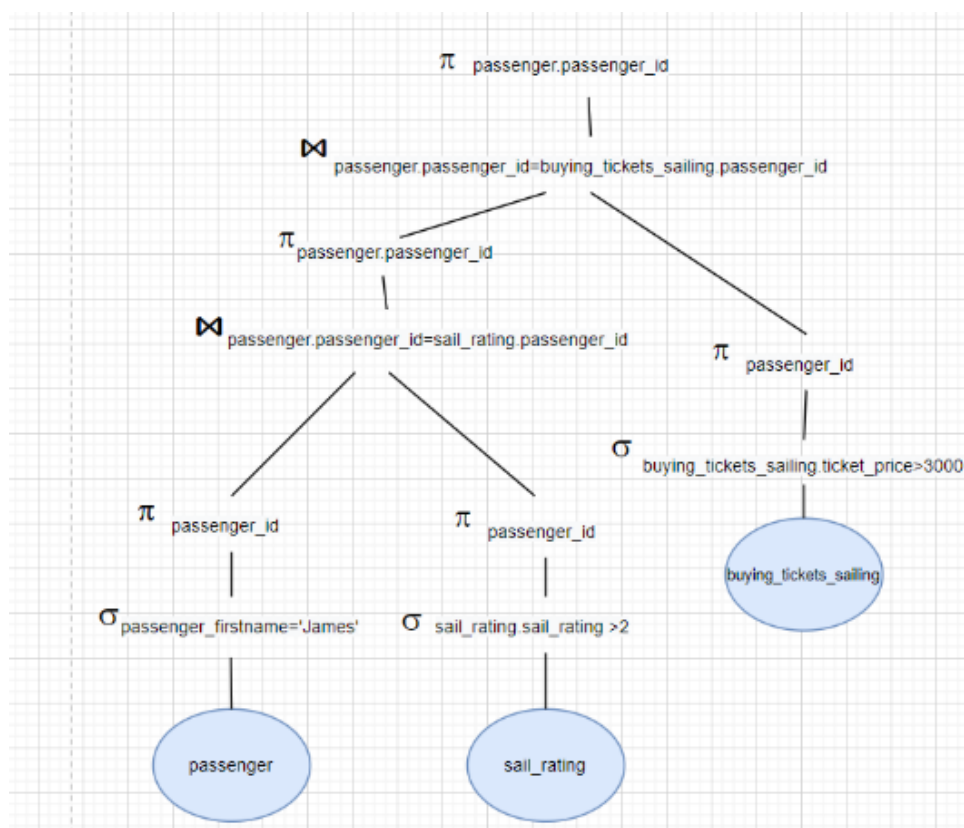


2. שלוף את כל תעודות הזהות של הנוסעים ששם הפרטי הוא גיימס שנתנו דירוג מעל ל-2 ומחיר הכרטיס ששילמו גדול מ-3000.

עץ שאילתה ראשוני



עץ שאילתה אופטימאלי



שאלה 12:

שלוף את כל תעודות הזהות של הנוסעים ששם הפרטי הוא גיימס שנתנו דירוג מעל ל-2 ומחיר הכרטיס ששילמו גדול מ-3000.

הנחות:

- גודל בלוק B500
- גודל כל שדה B10
- זמן פעולת I/O 10ms

מהנתונים שהכנסתי:

- בטבלה passenger 25000 רשומות ו 5 שדות (יש 367 רשומות ששם הפרטי הוא 'james')
- בטבלה sail_rating 77635 רשומות ו 3 שדות (יש 46784 רשומות שנתנו נוסעים דירוג מעל 2)
- בטבלה buying_tickets_sailing 77635 רשומות ו 4 שדות (יש 53901 רשומות שמחיר הכרטיס ששילמו נוסעים גדול מ3000)

זמן השאילתה הצפוי לפי עץ השאילתה האופטימלי ללא האינדקס:

עבור join ראשון בין ship ל company:

- $B_{passenger} - (נצטרך 1 שדות), ceil(1*10*367/500)=8$
- $B_{sail_rating} - (נצטרך 1 שדות), ceil(1*10*46784/500)=936$

שימוש ב hash join הראשון:

- $3*(8+936)=2832$

עבור join השני sail לבין join של ship , company:

- $B_{buying_tickets_sailing} - (נצטרך 1 שדות), ceil(1*10*53901/500)=1079$

שימוש ב hash join השני:

- $3*(936+1079)=6045$

תוצאה: I/O 2832+6045=8877

$$8877*10/1000/60=1.4795 \text{ minutes}$$

ב.זמן השאילתה הצפוי לאחר הגדרת האינדקס :

ניתן לשפר את זמן הריצה באמצעות אינדקס ערבול על passenger_id בטבלת sail_rating.

- יש 77635 רשומות של ציונים לpassenger25000 כלומר כל נוסע מדרג בממוצע 3.1 פעמים.
בהנחה שהאינדקס אינו מקובץ יעלה לנו ה join הראשון:
 $8+(1.2+3.1)*367=1587$ I/O

ניתן לשפר את זמן הריצה באמצעות אינדקס ערבול על passenger_id בטבלת passenger_tickets_sailing.

- יש 77635 רשומות של קניית כרטיסים של נוסעים לpassenger25000 כלומר כל נוסע מפליג בממוצע 3.1 פעמים.
בהנחה שהאינדקס אינו מקובץ יעלה לנו ה join השני:
 $8+(1.2+3.1)*367=1587$ I/O

תוצאה : $1587+1587=3174$ I/O

$3174*10/1000/60=0.529$ minutes

שאלה 13:

לכל עובד שלוף את שכר העובד ומזהה הספינה וצור עמודה חדשה שמביאה את השכר של העובד במקום החמישי לפניו לפי סדר שכר העובדים בכל ספינה.

השאלתה תחלק לחלוקות לפי ship_id לכל ספינה ותמיינ את שכר העובדים בכל חלוקה ותכניס לעמודה חדשה את השכר של העובד במקום החמישי לפניו.

```

1 select employee_salary,ship_id,
2 LAG(employee_salary,5)
3 OVER
4 (PARTITION BY ship_id
5 ORDER BY employee_salary)AS Prev_employee_salary
6 from employee

```

Explain Notifications Messages Data Output Scratch Pad

	employee_salary integer	ship_id integer	prev_employee_salary integer
1	750	1000	[null]
2	2000	1000	[null]
3	3000	1000	[null]
4	3000	1000	[null]
5	3000	1000	[null]
6	3000	1000	750
7	3500	1000	2000
8	3500	1000	3000
9	4000	1000	3000
10	4000	1000	3000
11	4000	1000	3000
12	5500	1000	3500

	employee_salary integer	ship_id integer	prev_employee_salary integer
92	10500	1003	9500
93	11000	1003	9500
94	12000	1003	10000
95	13500	1003	10000
96	15000	1003	10500
97	16000	1003	10500
98	18000	1003	11000
99	19500	1003	12000
100	27500	1003	13500
101	1000	1008	[null]
102	2000	1008	[null]
103	3000	1008	[null]
104	3000	1008	[null]
105	3500	1008	[null]
106	4000	1008	1000
107	4000	1008	2000

	employee_salary integer	ship_id integer	prev_employee_salary integer
6949	18000	1342	13000
6950	20000	1342	13000
6951	500	1343	[null]
6952	500	1343	[null]
6953	3000	1343	[null]
6954	3000	1343	[null]
6955	3000	1343	[null]
6956	3500	1343	500
6957	3500	1343	500
6958	3500	1343	3000
6959	4000	1343	3000
6960	4000	1343	3000
6961	4000	1343	3500
6962	4000	1343	3500
6963	4000	1343	3500

שאלה 14:

```

1
2 select * from (
3 select e2.employee_salary,e2.ship_id,e1.employee_salary as Prev_employee_salary from(
4 select employee_id,employee_salary,ship_id, row_number() OVER (PARTITION BY ship_id ORDER BY employee_salary)
5 from employee) as e1
6 join (
7 select employee_id,employee_salary,ship_id, row_number() OVER (PARTITION BY ship_id ORDER BY employee_salary)
8 from employee) as e2
9 on e1.ship_id=e2.ship_id and e1.row_number+5=e2.row_number
10 union all
11 select employee_salary,ship_id,null as Prev_employee_salary from (
12 select *, row_number() OVER (PARTITION BY ship_id ORDER BY employee_salary) AS ranka
13 FROM employee
14 ) as ma_1
15 where ma_1.ranka<6
16 ) as n
17 order by n.ship_id,n.employee_salary, - n.prev_employee_salary desc
18

```

Explain Notifications Messages Data Output Scratch Pad

	employee_salary integer	ship_id integer	prev_employee_salary integer
1	750	1000	[null]
2	2000	1000	[null]
3	3000	1000	[null]
4	3000	1000	[null]
5	3000	1000	[null]
6	3000	1000	750
7	3500	1000	2000
8	3500	1000	3000

	employee_salary integer	ship_id integer	prev_employee_salary integer
3690	10000	1187	8500
3691	10500	1187	8500
3692	11000	1187	9000
3693	11000	1187	9000
3694	11500	1187	10000
3695	12500	1187	10000
3696	13000	1187	10500
3697	16000	1187	11000
3698	16500	1187	11000
3699	19500	1187	11500
3700	22150	1187	12500
3701	1250	1191	[null]
3702	1750	1191	[null]
3703	3000	1191	[null]
3704	3000	1191	[null]
3705	3500	1191	[null]
3706	3500	1191	1250
3707	3500	1191	1750
3708	4000	1191	3000
3709	4000	1191	3000

	employee_salary integer	ship_id integer	prev_employee_salary integer
8789	10000	1429	8500
8790	10500	1429	8500
8791	11000	1429	8500
8792	12000	1429	9000
8793	12500	1429	9500
8794	13000	1429	10000
8795	13000	1429	10500
8796	15000	1429	11000
8797	16000	1429	12000
8798	16500	1429	12500
8799	19000	1429	13000
8800	25000	1429	13000
8801	750	1431	[null]
8802	1000	1431	[null]
8803	3000	1431	[null]
8804	3000	1431	[null]
8805	4000	1431	[null]
8806	4000	1431	750
8807	4000	1431	1000
8808	4000	1431	3000
8809	4000	1431	3000

שאלה 15:

אנו מעוניינים שלפני כל הכנסה של עובד חדש, כל העובדים ששייכים לאותה ספינה שהעובד החדש יעבוד בה מקבלים בונוס של העלאה למשכורת של 1000 שקל.

```
CREATE OR REPLACE FUNCTION trigger()
RETURNS TRIGGER
LANGUAGE PLPGSQL
AS $$
DECLARE
bonus integer = 1000;









BEGIN
update employee
set employee_salary = bonus + employee_salary
where ship_id = new.ship_id;
return new;

END;
$$;
CREATE TRIGGER commission BEFORE INSERT ON employee for each row EXECUTE FUNCTION trigger();
INSERT into employee(
employee_id,employee_firstname,employee_lastname,employee_birthday,employee_salary,employee_id_captain,ship_id)
VALUES (208447029, 'Matan', 'Leventer', '21-10-1996',1000,798654621,1369);
```

לפני הכנסה:

	employee_id [PK] integer	employee_firstname character varying (50)	employee_lastname character varying (50)	employee_birthday date	employee_salary integer	employee_id_captain integer	ship_id integer
36	533176488	Christopher	Black	1974-08-17	6500	47699949	1369
37	349889906	Michael	Ross	1986-04-10	6500	54009534	1369
38	563450232	Lydia	Kim	1982-05-03	5500	534416799	1369
39	337170795	Candice	Wilson	1987-10-24	4500	374547977	1369
40	453277809	Brittany	Mullins	1979-01-01	4000	47699949	1369
41	861637727	Adam	Kelley	1977-01-22	3500	599604648	1369
42	873886892	Matthew	Nguyen	1984-12-12	3500	534416799	1369
43	713736812	Ann	Cooper	1980-12-29	3500	534416799	1369
44	436033127	Wesley	Yates	1985-06-11	3500	534416799	1369
45	312380296	Tiffany	Lewis	1971-02-26	3000	828488622	1369
46	795778216	Gail	Bradley	1988-06-23	3000	374547977	1369
47	712776016	Katherine	Davidson	1986-11-06	3000	54009534	1369
48	272556361	Courtney	Howard	1971-05-13	3000	374547977	1369
49	800003606	Jared	Orozco	1974-01-14	1250	798654621	1369
50	199521911	Karen	Gonzalez	1980-10-18	750	798654621	1369

לאחר הכנסה:

	 employee_id [PK] integer 	employee_firstname character varying (50) 	employee_lastname character varying (50) 	employee_birthday date 	employee_salary integer 	employee_id_captain integer 	ship_id integer 
37	349889906	Michael	Ross	1986-04-10	7500	54009534	1369
38	563450232	Lydia	Kim	1982-05-03	6500	534416799	1369
39	337170795	Candice	Wilson	1987-10-24	5500	374547977	1369
40	453277809	Brittany	Mullins	1979-01-01	5000	47699949	1369
41	713736812	Ann	Cooper	1980-12-29	4500	534416799	1369
42	436033127	Wesley	Yates	1985-06-11	4500	534416799	1369
43	873886892	Matthew	Nguyen	1984-12-12	4500	534416799	1369
44	861637727	Adam	Kelley	1977-01-22	4500	599604648	1369
45	312380296	Tiffany	Lewis	1971-02-26	4000	828488622	1369
46	272556361	Courtney	Howard	1971-05-13	4000	374547977	1369
47	795778216	Gail	Bradley	1988-06-23	4000	374547977	1369
48	712776016	Katherine	Davidson	1986-11-06	4000	54009534	1369
49	800003606	Jared	Orozco	1974-01-14	2250	798654621	1369
50	199521911	Karen	Gonzalez	1980-10-18	1750	798654621	1369
51	208447029	Matan	Leventer	1996-10-21	500	798654621	1369

שאלה 16:

רשות הספינות העולמית החליטה שמותר לצאת להפלגה רק פעם אחת ביום ולכן נצטרך לבדוק האם יש לנו הפלגות באותו תאריך ולעדכן בהתאם הפלגות אלו בימים אחרים. הפקודה רשומה בפייתון.

```
7
8 select *
9 from sail
```

	sail_id [PK] integer	port_id_enter integer	port_id_exit integer	sail_date date	ship_id integer
1	23904	315	331	2016-07-29	1276
2	56341	370	393	2018-05-27	1074
3	76578	308	357	2016-10-10	1362
4	12190	364	354	2011-10-06	1063
5	59447	391	307	2011-07-15	1134
6	30325	337	389	2019-12-14	1077
7	83697	391	311	2011-07-07	1322
8	10465	307	341	2013-11-17	1169
9	70538	365	377	2017-07-02	1449
10	26111	377	319	2013-03-22	1206
11	51596	368	355	2015-12-10	1386
12	45043	369	365	2018-10-25	1155
13	29336	373	312	2020-01-28	1073
14	28580	341	354	2020-07-02	1175
15	67209	311	337	2014-06-25	1195

✓ Successfully run. Total query runtime: 149 msec. 1000 rows affected.

```
7
8 select distinct(sail_date)
9 from sail
```

	sail_date date
1	2011-06-24
2	2012-06-08
3	2016-03-27
4	2013-07-17
5	2020-08-29
6	2016-01-13
7	2010-10-11
8	2012-10-18
9	2018-11-09
10	2015-01-23
11	2014-04-03
12	2015-10-10
13	2016-11-24
14	2010-12-22
15	2011-10-14
16	2020-12-26

✓ Successfully run. Total query runtime: 118 msec. 884 rows affected.

```
1 select distinct(sail_date) from sail
```

	sail_date date
1	2012-06-08
2	2018-11-05
3	2013-07-17
4	2020-08-29
5	2016-01-13
6	2010-10-11
7	2014-04-03
8	2015-10-10
9	2019-10-20
10	2010-03-15
11	2011-02-14
12	2010-12-22
13	2010-06-14
14	2011-11-05
15	2015-07-20

✓ Successfully run. Total query runtime: 164 msec. 1000 rows affected.

ניתן לראות שבטבלת Sail יש 1000 רשומות. אולם יש רק 884 תאריכים שונים, ולכן נצטרך לעדכן בהתאם את תאריכי הפלגות, לתאריכים אחרים. הפקודה רשומה בקובץ רשומה בקובץ פייתון.

ניתן לראות שלאחר הרצת הפקודה בפייתון. שכעת יש 1000 תאריכים שונים, כלומר אין 2 הפלגות שיוצאות באותו יום.