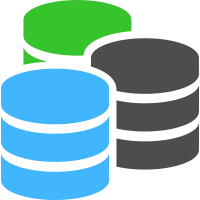
בס"ד , תשע"ט 2019

מיני פרויקט בסיסי נתונים

חלק ב' group A+D



מגישים:

אלעד אליהו :203527379

יוסף קראי : 308161447

תוכן עניינים

[2.0 תיאור האגפים A ו-D והקשר בניהם 4](#_Toc12409132)

[2.0.1 אגף A מחלקת משאבי אנוש 4](#_Toc12409133)

[תיאור כללי (לקוח מקבוצה A) 4](#_Toc12409134)

[ERD Diagram 8](#_Toc12409135)

[2.0.2 אגף D מחלקת רכב 9](#_Toc12409136)

[תיאור כללי (לקוח מקבוצה D) 9](#_Toc12409137)

[תיאור של ישויות 9](#_Toc12409138)

[תיאור של קשרים 10](#_Toc12409139)

[ERD Diagram 11](#_Toc12409140)

[2.0.3 DSD diagram של שתי המחלקות 12](#_Toc12409141)

[DSD diagram של מחלקת רכב E 12](#_Toc12409142)

[DSD diagram של מחלקת משאבי אנוש E 12](#_Toc12409143)

[2.1 איחוד האגפים A+D 13](#_Toc12409144)

[2.1.0 תמצית 13](#_Toc12409145)

[2.1.1 Import של מחלקת משאבי אנוש A 13](#_Toc12409146)

[2.1.2 שינוי השם של טבלאות הגרעין ב- A ל NAMETemp 14](#_Toc12409147)

[2.1.3 Import של מחלקת רכב D 14](#_Toc12409148)

[2.1.4 יישור קו בין טבלאות של הגרעין 15](#_Toc12409149)

[2.1.5 מיזוג טבלאות הגרעין 15](#_Toc12409150)

[מעבר מ areatemp ל- area 16](#_Toc12409151)

[מעבר מ- citytemp ל- city 17](#_Toc12409152)

[מעבר מ- policemamtemp ל- policemam 18](#_Toc12409153)

[מעבר מ- citizentemp ל- citizen 20](#_Toc12409154)

[מעבר מ- scheduletemp ל- schedule 21](#_Toc12409155)

[2.1.6 הפניית FK למקומות המתאימים 24](#_Toc12409156)

[2.1.7 מחיקת טבלאות ה- temp 25](#_Toc12409157)

[2.1.8 הקשר בין A ל- D 26](#_Toc12409158)

[הקשר שעשינו: 26](#_Toc12409159)

[יצירת הטבלה 26](#_Toc12409160)

[איכלוס הטבלה 27](#_Toc12409161)

[3.0 שאילתות SQL 28](#_Toc12409162)

[3.1 SQL Queries 28](#_Toc12409163)

[שאילתא 1 28](#_Toc12409164)

[שאילתא 2 28](#_Toc12409165)

[שאילתא 3 29](#_Toc12409166)

[שאילתא 4 30](#_Toc12409167)

[שאילתא 5 30](#_Toc12409168)

[שאילתא 6 31](#_Toc12409169)

[שאילתא 7 32](#_Toc12409170)

[שאילתא 8 33](#_Toc12409171)

[3.2 יצירת אינדקסים 34](#_Toc12409172)

[4.0 פקודות Alter 36](#_Toc12409173)

[5.0 עדכון בסיס הנתונים – DML 38](#_Toc12409174)

[5.1 פקודות Update 38](#_Toc12409175)

[Update 1 - הדגמה של rollback & commit 38](#_Toc12409176)

[Update 2 - חוצה אגפים 41](#_Toc12409177)

[5.2 פקודות Delete 43](#_Toc12409178)

[Delete 1 - חוצה אגפים 43](#_Toc12409179)

[Delete 2 45](#_Toc12409180)

[6.0 יצירת מבטים - Views 46](#_Toc12409181)

[View 1 46](#_Toc12409182)

[View 2 48](#_Toc12409183)

[View 3 50](#_Toc12409184)

[7.0 תוכנית PL-SQL 52](#_Toc12409185)

[Program 1 52](#_Toc12409186)

[Program 2 54](#_Toc12409187)

[הוספת trigger 58](#_Toc12409188)

[Program 3 59](#_Toc12409189)

# 2.0 תיאור האגפים A ו-D והקשר בניהם

## 2.0.1 אגף A מחלקת משאבי אנוש

### **תיאור כללי** (לקוח מקבוצה A)

**Tables**

**AREA**

Description

An area where the Police operates. An area has many cities.

Fields

* AreaID (Number): A unique identifier that represents the area.
* AreaName (String): The name of the area.

Keys

* AreaID (Primary)

**CITIZEN**

Description

A citizen that lives in a city, and some informations about him.

Fields

* CitizenID (Number): A unique identifier that represents the citizen.
* CitizenName (String): The name of the citizen.
* CityName (String): The city where the citizen lives.
* Address (String): The address of the citizen.
* PhoneNR (String): The phone number of the citizen.

Keys

* CitizenID (Primary)
* CityName (Foreign: City)

**CITY**

Description

A city where the police operates. Every city is in an area.

Fields

* CityName (String): A unique identifier for the city.
* AreaID (Number): A reference to the area where the city is.

Keys

* CityName (Primary)
* AreaID (Foreign: Area)

**CONTRACT**

Description

A contract between a policeman and a station. Contains various informations, like start date and extra salary.

Fields

* ContractID (Number): A unique identifier for the contract.
* StartDate (Date): Contract date.
* EndDate (Date?): Contract expiration date.
* ExtraSalary (Number): An extra that the policeman gets at the end of the year. Can be zero.
* Role (String): The role of the policeman. Can take any of the values : [‘FieldAgent’, ‘OfficeAgent’, ‘Cadre’]
* PolicemanID (Number): A reference to the policeman.
* StationID (Number): A reference to the station.

Keys

* ContractID, PolicemanID, StationID (Primary)
* PolicemanID (Foreign: Policeman)
* StationID (Foreign: Station)

**GARAGE**

Description

A garage for the Police.

Fields

* GarageID (Number): A unique identifier for the garage.
* Capacity (Number): The capacity, in square meters, of the garage. Can take any of the values: [200, 500, 1000]
* WarehouseID (Number?): A reference to the warehouse that contains the garage.

Keys

* GarageID (Primary)
* WarehouseID (Foreign: Warehouse)

**POLICEMAN**

Description

A policeman, and many informations about him.

Fields

* PolicemanID (Number): A unique identifier for the policeman.
* PolicemanName (String): The name of the policeman.
* Rating (Number): Rating of the policeman. Ranges between 1 and 5.
* HireYear (Date): The date when the policeman started working.
* BossID (Number): A reference to his boss, who is another policeman. If equal to his PolicemanID, it means that he has no boss.
* Salary (Number): The salary of the policeman. Can take any of the values: [7500, 10000, 15000]
* AreaID (Number): A reference to the area where the policeman works.

Keys

* PolicemanID (Primary)
* AreaID (Foreign: Area)

**PROMOTIONTRACK**

Description

The current position that the policeman occupies.

Fields

* Status (String): The role of the policeman.
* PolicemanID (Number): A reference to a policeman.

Keys

* PolicemanID (Foreign: Policeman)

**SCHEDULE**

Description

An event involving a policeman and a citizen.

Fields

* Datameeting (Number): A unique identifier for the event.
* CitizenID (Number): A reference to a citizen.
* PolicemanID (Number): A reference to a policeman.

Keys

* Datameeting (Primary)
* CitizenID (Foreign: Citizen)
* PolicemanID (Foreign: Policeman)

**STATION**

Description

A Police station.

Fields

* StationID (Number): A unique identifier for the station.
* StationName (String): The name of the station.
* CityName (String): A reference to the city where the station is.

Keys

* StationID (Primary)
* CityName (Foreign: City)

**WAREHOUSE**

Description

A warehouse for a station

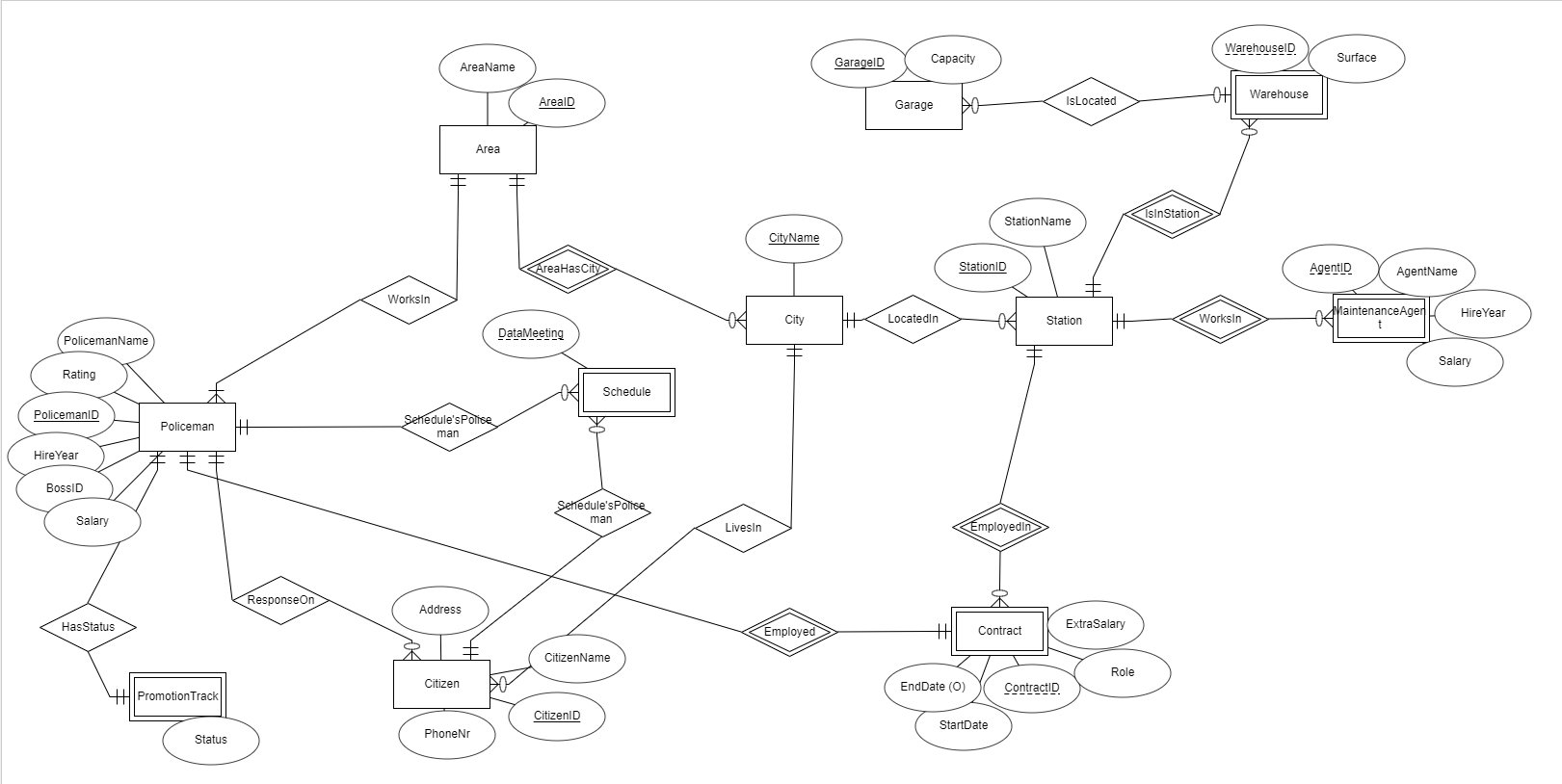
Fields

* WarehouseID (Number): A unique identifier for the warehouse.
* Surface (Number): The surface of the warehouse. Can take any of the values : [200, 500, 1000, 2000]
* StationID (Number): A reference to the station to which the warehouse belongs.

Keys

* WarehouseID (Primary)
* StationID (Foreign: Station)

### ERD Diagram



## 2.0.2 אגף D מחלקת רכב

### **תיאור כללי** (לקוח מקבוצה D)

In the vehicles department we will manage the police vehicles.

We will connect police officers to vehicles.

We have suppliers who supply vehicles by means of contracts signed between them and the police

We also have garages, so that each vehicle belongs to a garage located in a particular area

### **תיאור של ישויות**

**Area**: contains an area name, and unique area ID.

**City**: contains a city name and an area ID of which the city is.

**Citizen**: a citizen who the police deals with. Contains a unique ID, a cityzen name, a phone number, address where the citizen lives, and name of his city.

**Schedule**: for scheduling meetings between policeman and citizen. Contains meeting date, ID of policeman and ID of cityzen.

**Policeman**: represents an internal agent of the police. Contains a unique police ID, a police name, a rating (from 1 to 100), current salary, the ID of area where worker works, boss ID (also an policeman).

**Vehicle:** the vehicles from police, contain a unique vehicle ID, a company name of the vehicle, the model name of the vehicle, a garage ID and a contract ID.

**Policemancar**: a inheritance from policeman that connected with vehicle, policemancar inherits all the attributes of policeman because he is also a policeman and has an additional attribute ifdriver.

**Supplier**: He sells vehicles for the police, contains a unique supplier ID, a supplier name and a category(cars, Trucks….)

**Contract**: a contract from a supplier for vehicles for the police, contains a contract ID, a date sign contract, a number of vehicle that sells and supplier ID.

**Garages**: The garage handles vehicles and they are connected to a certain area, contain a garages ID, a garages name and a area ID.

**VehicleInGarages**: vehicleingarages Linking vehicles to the garage, contain a Treatment ID, a date that begin the Treatment, a date that end the Treatment, a vehicle ID and a garages ID.

### **תיאור של קשרים**

**areahascity**- one-to-many- relation that gives every city an area

**livein**- one-to-many- relation that gives every citizen a city

**worksin-** one-to-many-relation between policeman and area. Describe the area every policeman is in.

**schedulepolicem**- one-to-many- relation between policeman and schedule. Describe the policeman in charge of the scheduled meeting.

**schedulepolicem -** one-to-many- relation between citizen and schedule. Describe the citizen in the scheduled meeting.

**responseon-** one-to-many- relation between citizen and policeman. Describe the supervisor policeman of group of citizens.

**inheritance**- one-to-one- an inheritance from policeman to policeman car.

**GaragesInArea**- one-to-many- relation between garages and area. Describe the area of the garage.

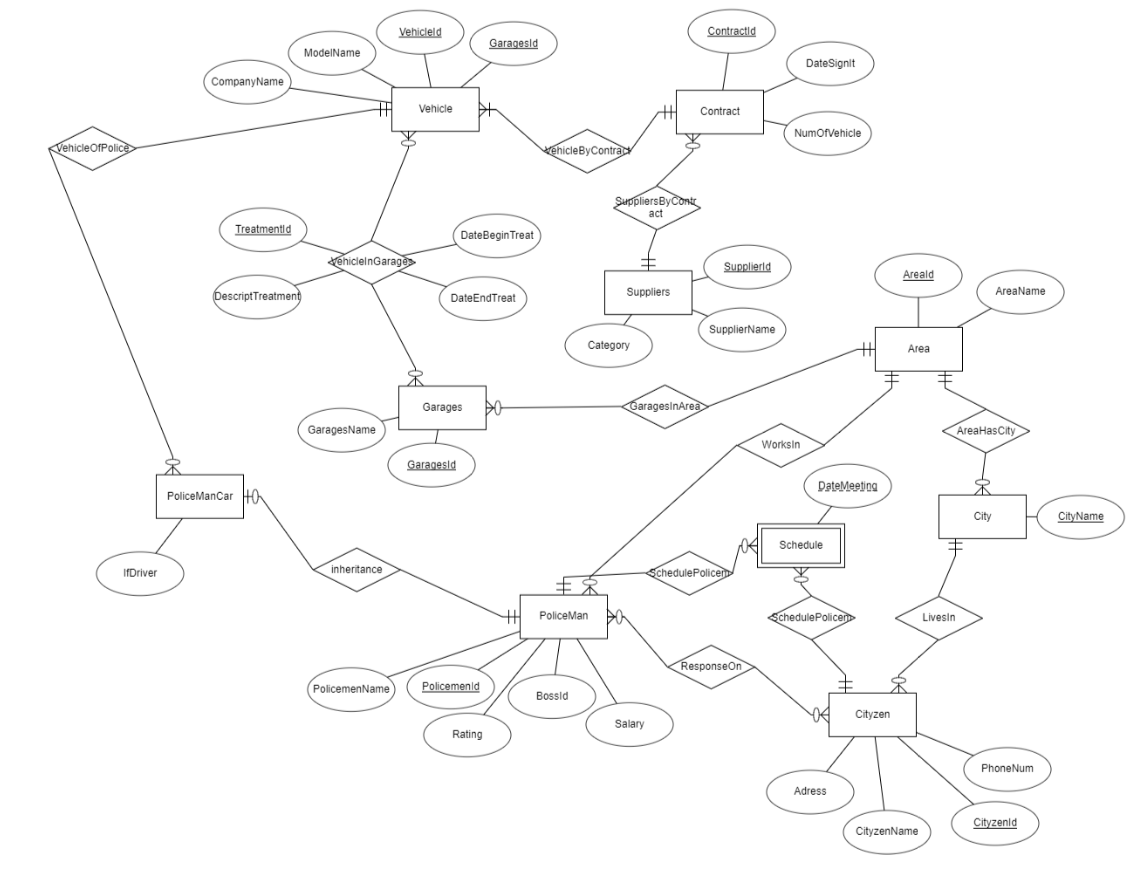
**VehicleOfPolice**- one-to-many- relation between policemancar and vehicle. Connected a policeman to a vehicl.

**VehicleByContract**- one-to-many- relation between vehicle and contract. Connected a vehicle to his contract.

**SuppliersByContract** - one-to-many- relation between supplier and contract. Describe the connected supplier with his contract.

**VehicleInGarages** - many-to-many- relation between vehicle and garages. Describe the garage that treat the vehicle.

### ERD Diagram



## 2.0.3 DSD diagram של שתי המחלקות

### **DSD diagram של מחלקת רכב E**

### **DSD diagram של מחלקת משאבי אנוש E**

אין

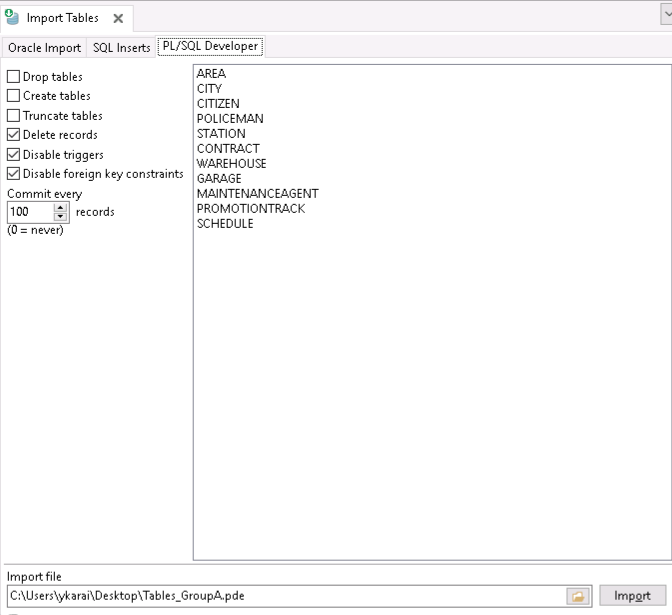
# 2.1 איחוד האגפים A+D

## 2.1.0 תמצית

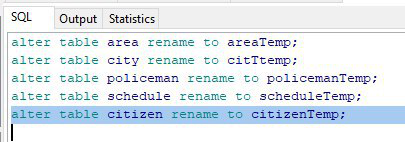
**שלב ראשון -** נאחד את הליבה המשותפת בצורה הבאה: -נייבא את מחלקת A ונשנה את שמות טבלאות הליבה לtemp, ואז נייבא את מחלקת D. -ניישר קו בין הטבלאות ואז נמזג את הליבה של מחלקת A לתוך הליבה של מחלקת D. -במחלקת A נשנה את הFK שיצביעו לליבה במחלקה D ואז אפשר למחוק את הליבה של D.

**שלב שני -** ניצור טבלה שמקשרת בין שני האגפים ונאכלס אותה בנתונים.

## 2.1.1 Import של מחלקת משאבי אנוש A

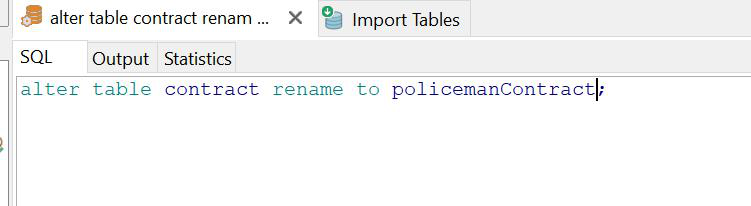


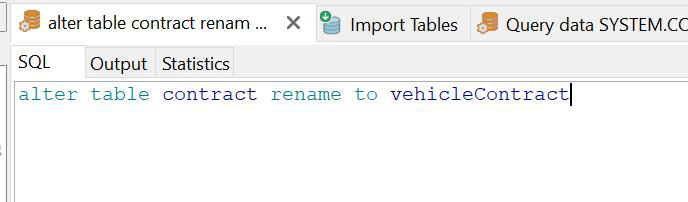
## 2.1.2 שינוי השם של טבלאות הגרעין ב- A ל NAMETemp



## C:\Users\ykarai\Downloads\d448a871-874d-44c7-875e-0e1336b7a0fb.jpg2.1.3 Import של מחלקת רכב D

**התנגשות בטבלה Contract**

בשתי הקבוצות יש את הישות contract לכן נשנה את השם בכל אחד מהם:



## 2.1.4 יישור קו בין טבלאות של הגרעין

כדי לא לאבד נתונים כאשר היה הבדל בין טיפוסים אזי הלכנו על שינוי כלפי הטיפוס הגדול, לדוגמא כאשר היה number(7) בטבלה אחת ו- integer בטבלה מקבילה אזי הלכנו כלפי ה- integer כיוון שהוא הטיפוס היותר גדול. אותו דבר לגבי varchar , נבחר המחרוזת הגדולה יותר..

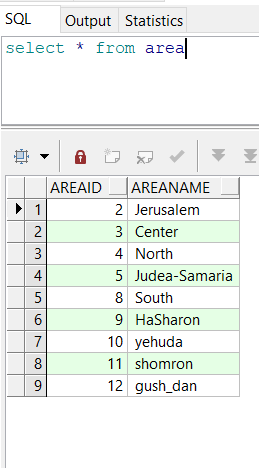
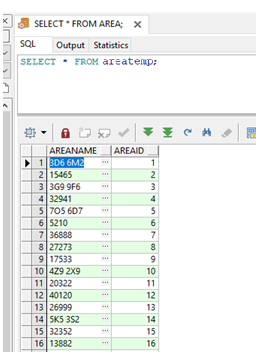
## 2.1.5 מיזוג טבלאות הגרעין

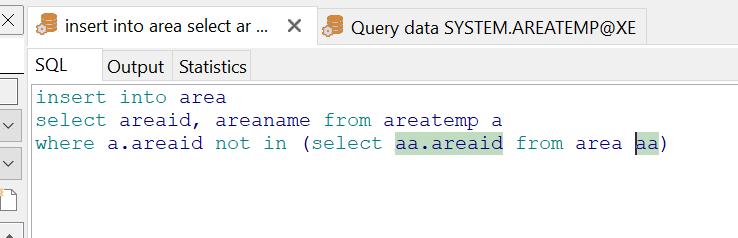
נעשה זאת לפי הסדר כדי למנוע בעיות עם מפתחות זרים.

הוחלט שאם בטבלאות הTemp יש מפתח שקיים כבר בטבלה המקבילה - לא נכניס את השורה הזאת (העיקר שיש את המפתח בטבלה וכך לא תהיה בעיה של ישויות המצביעות לאותו מפתח).

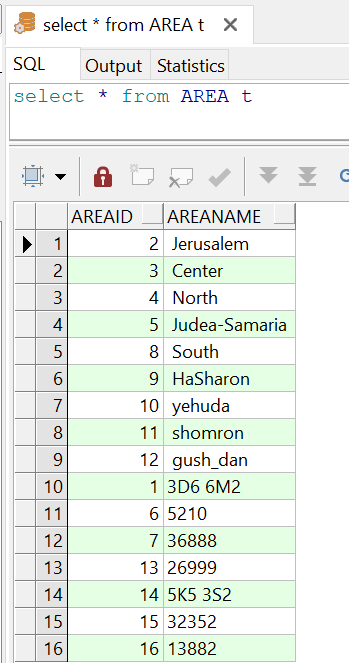
נראה את הטבלאות לפני ואח"כ המיזוג בהתאמה..

### **מעבר מ areatemp ל- area**

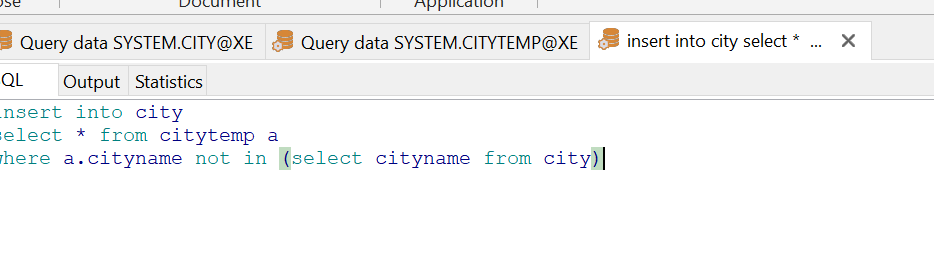
 



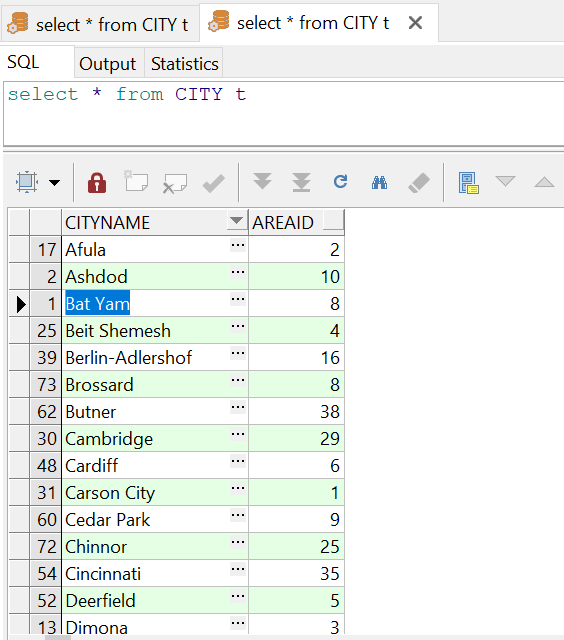
טבלה ממוזגת:



### **מעבר מ- citytemp ל- city**

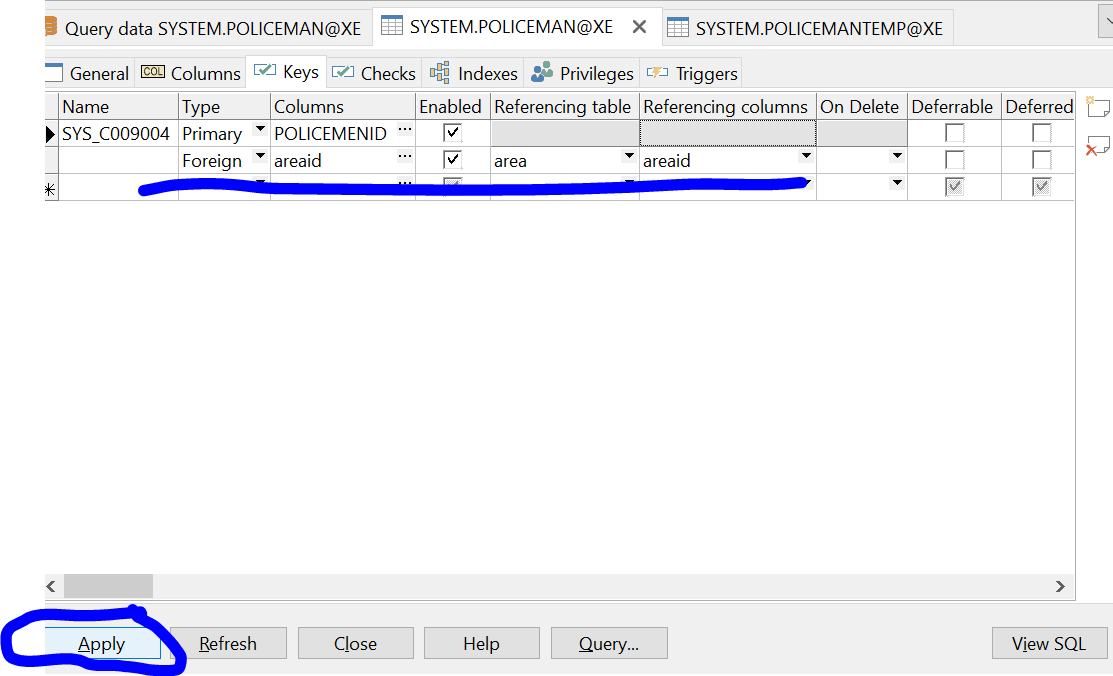


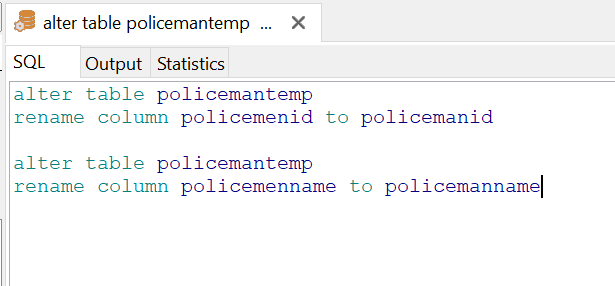
טבלה ממוזגת:



### **מעבר מ- policemamtemp ל- policemam**

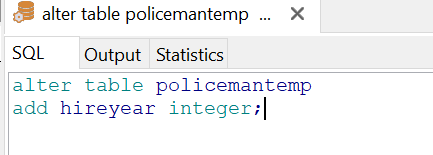
בpoliceman העמודה areaid לא מוגדר כמפתח זר לכן נגדיר את זה דרך "edit table":





שינוי שמות העמודות במקום policem**e**n לpolicem**a**n:

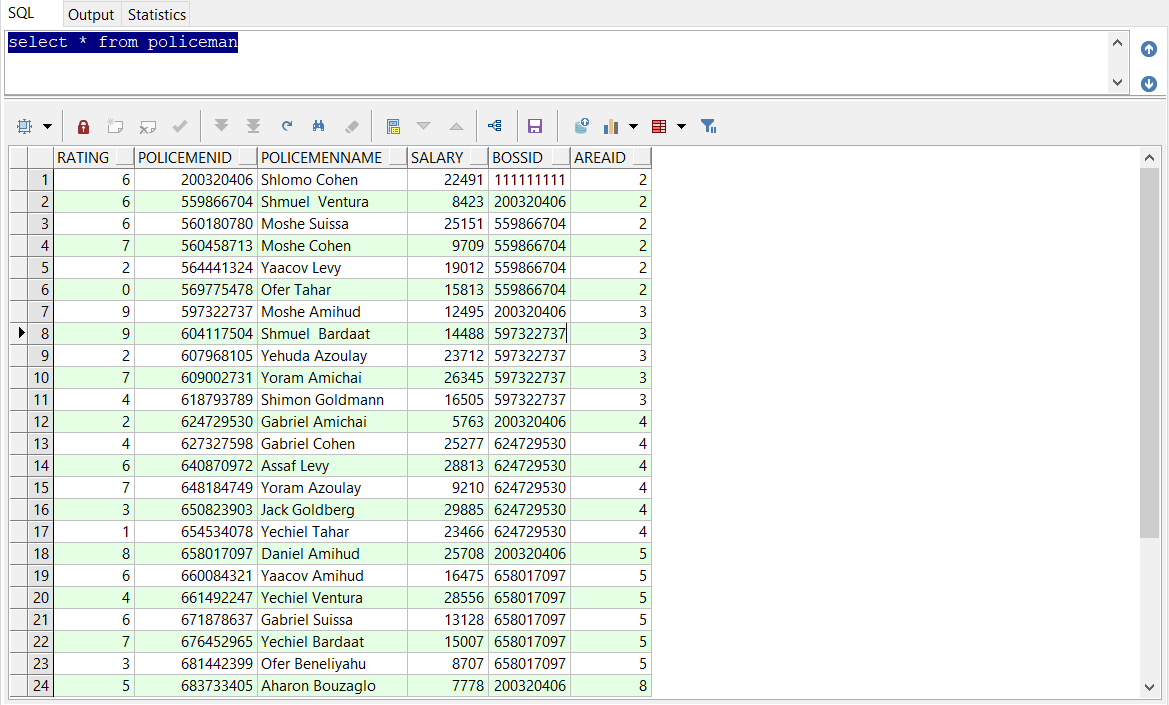
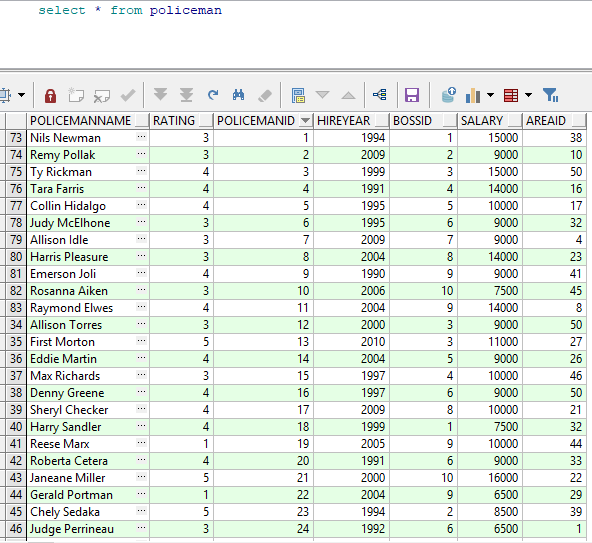
נוסיף עמודה hireyear לpolicemantemp :

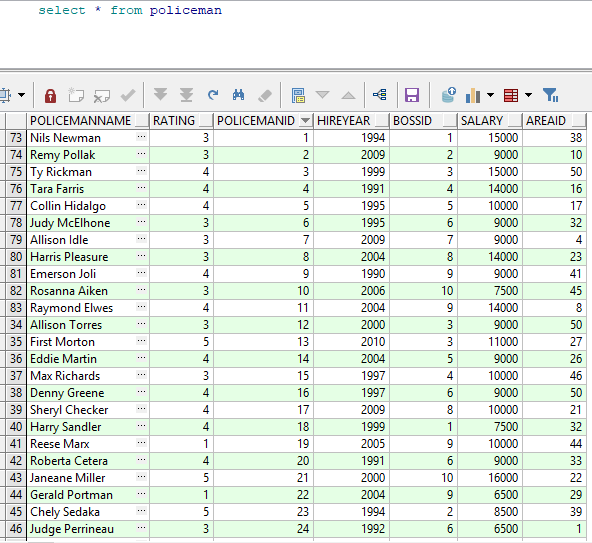


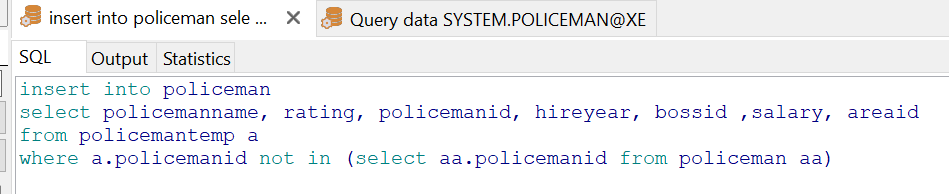
\*בנוסף, ב policeman עשינו אפשרות של nullable לעמודה hireyear (מאחר שבpolicemanTemp לא היתה עמודה זו)

[בשלב הסופי של הפרויקט נמלא את העמודה הזאת באמצעות תוכנית PLSQL.](#_Program_3)

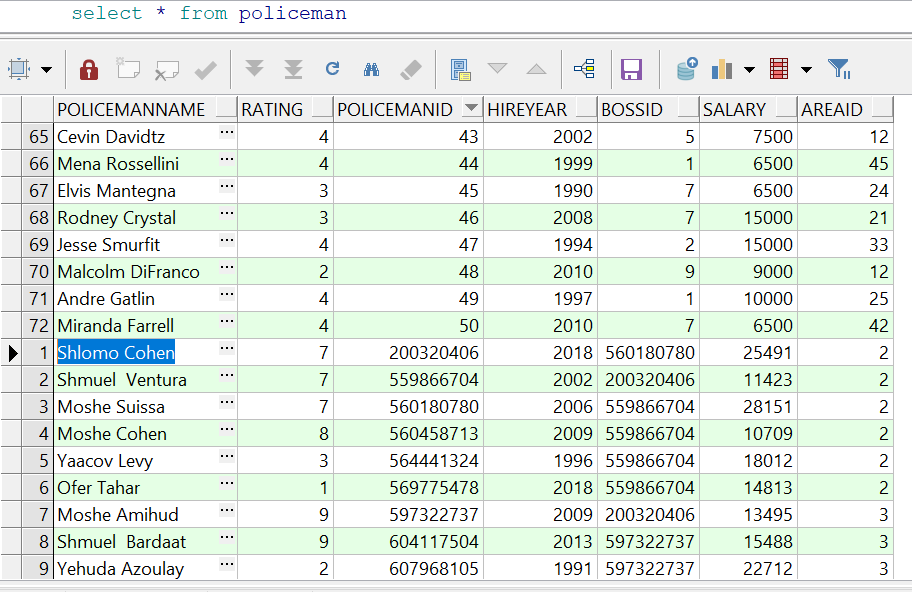
ועכשיו המיזוג:

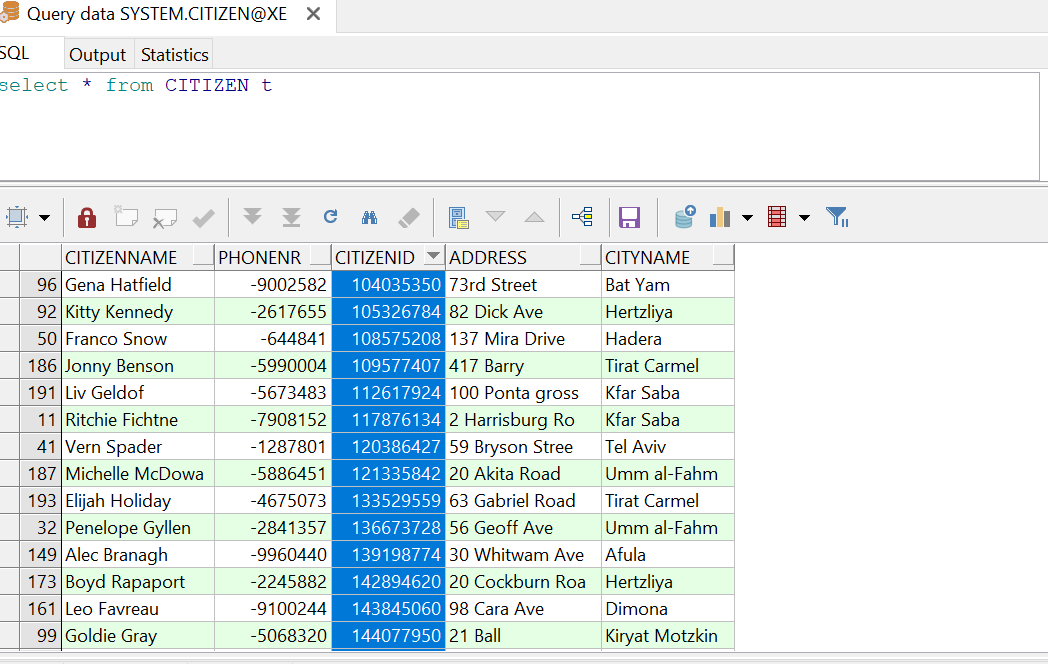


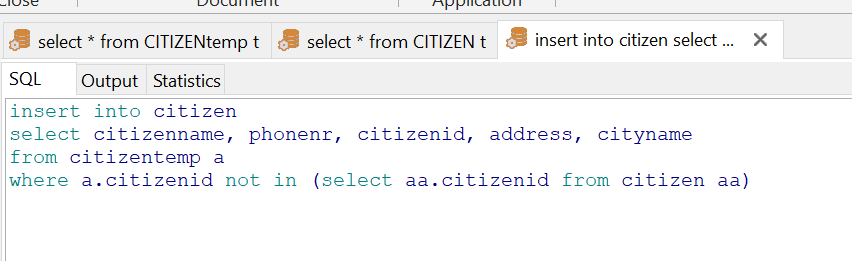


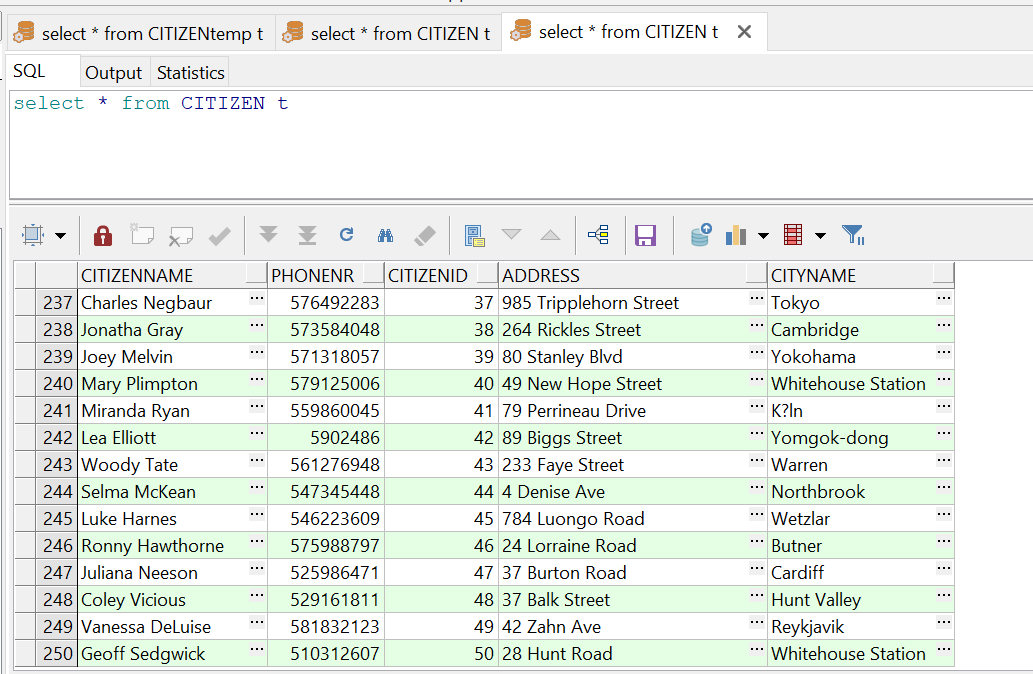
טבלה ממוזגת:



### **מעבר מ- citizentemp ל- citizen**

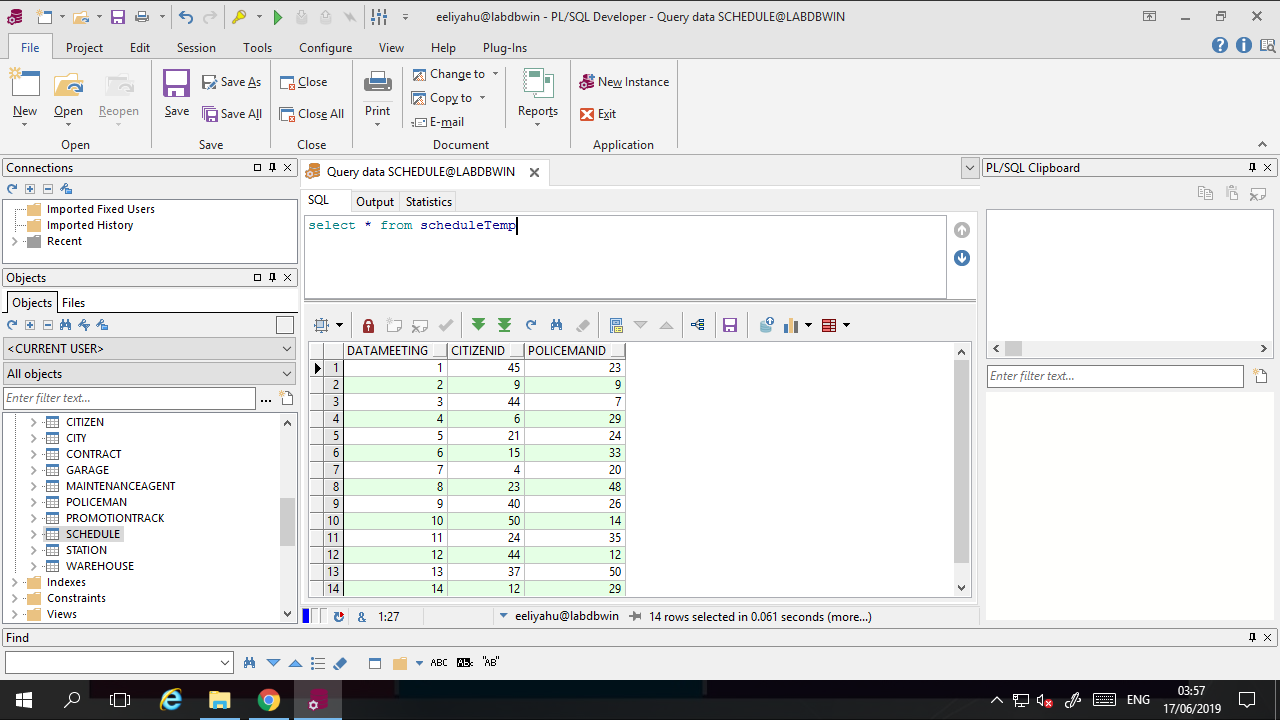




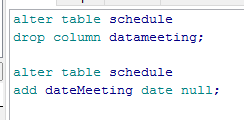


### **מעבר מ- scheduletemp ל- schedule**

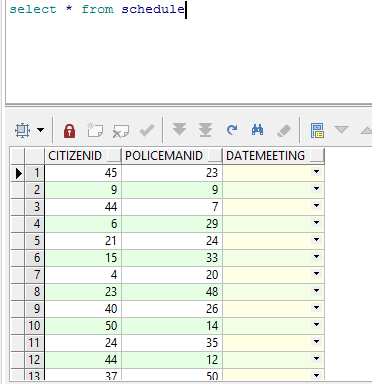
בטבלה scheduleTemp העמודה של התאריכים מוגדרת מסוג **integer**:



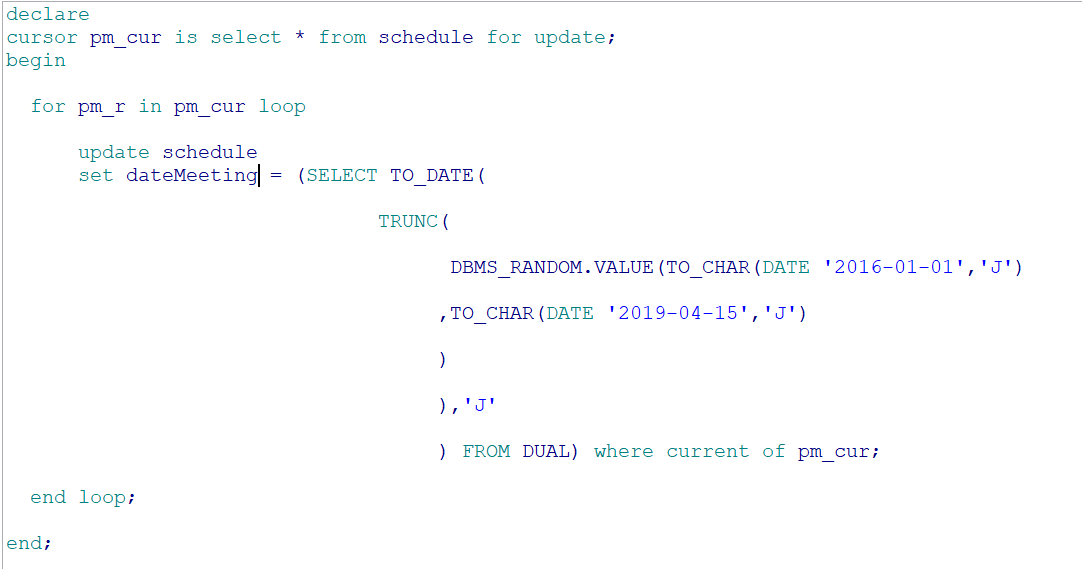
לכן מה שנעשה זה – נמחק את העמודה הקיימת וניצור עמודה חדשה מסוג **date**:

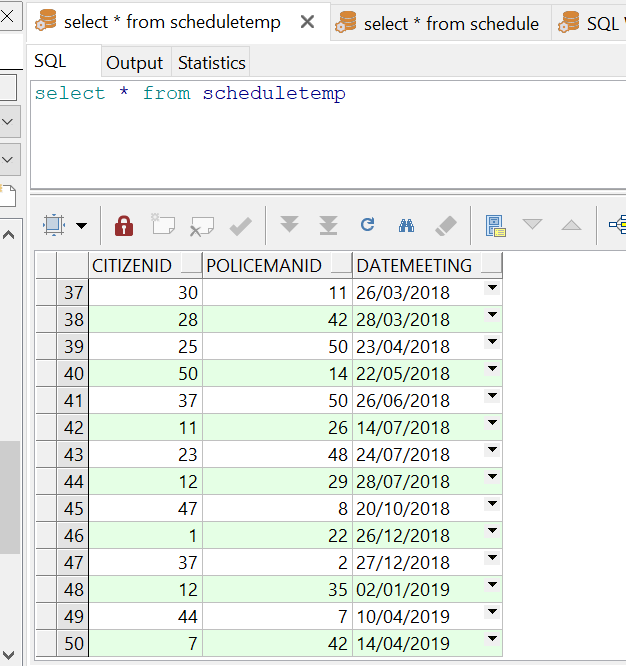


תוצאה:

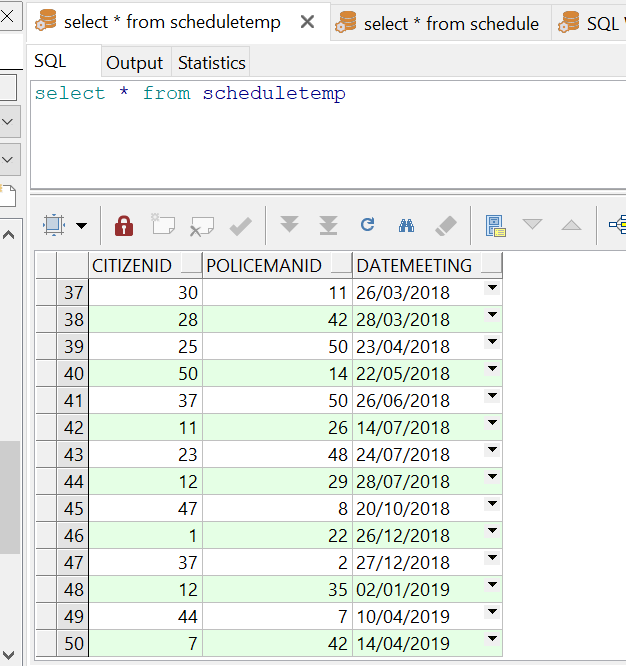
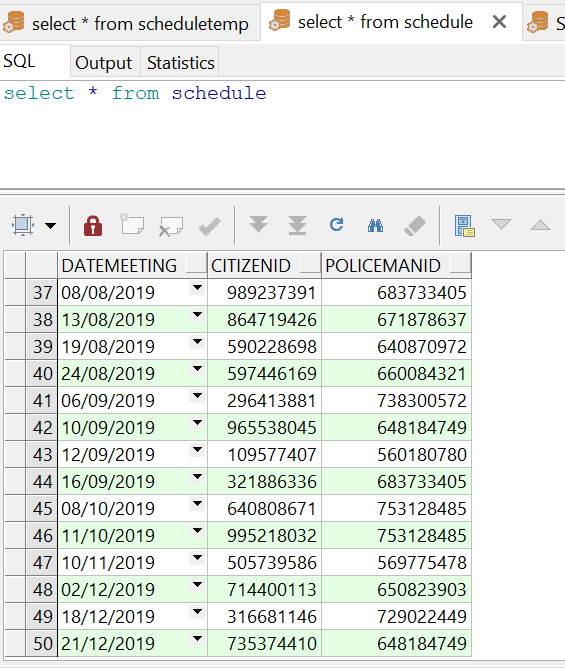


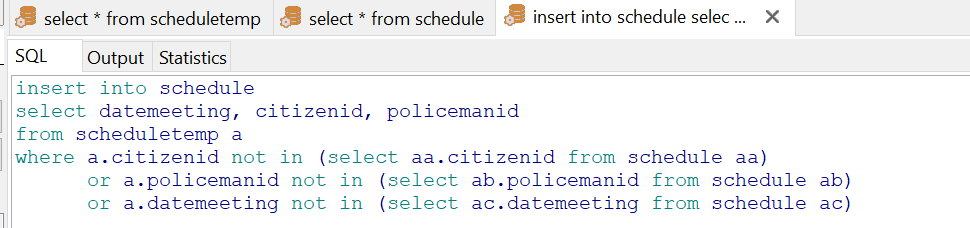
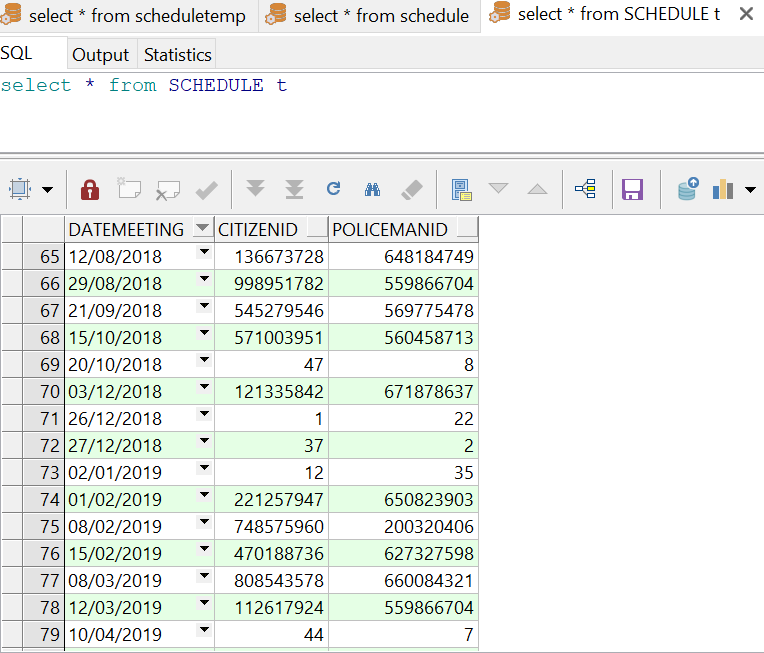
עכשיו נמלא את העמודה באמצעות תוכנית PL-SQL:



תוצאה:

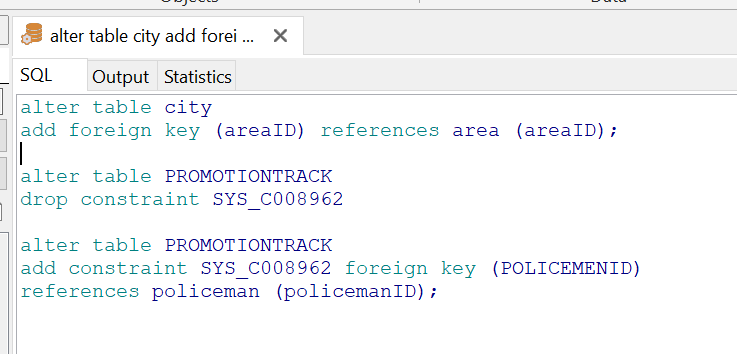
עכשיו המיזוג:



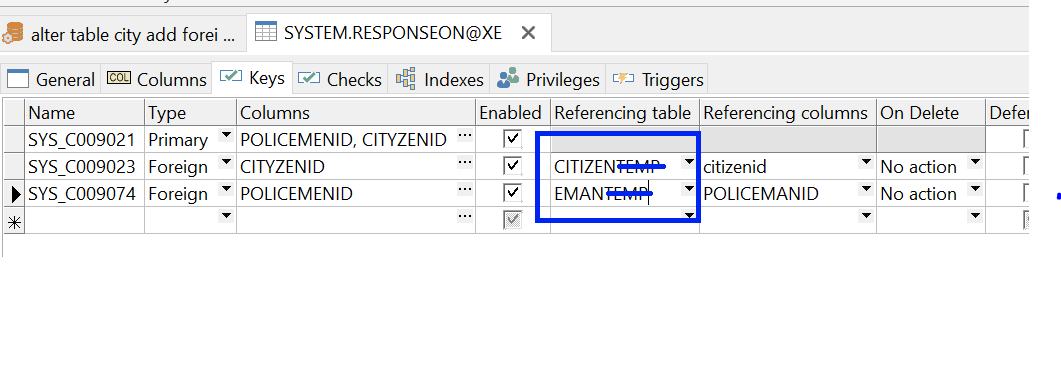


## 2.1.6 הפניית FK למקומות המתאימים

הטבלאות של הישויות וקשרים הנוספים במחלקה A - התייחסו לטבלאות ה- temp וע"כ נפנה את הקשרים האלו לטבלאות החדשות והמלאות.



מכיוון שיש הרבה FK לשנות, החלטנו לעשות את זה דרך הממשק הנוח של הPLSQL - פשוט למחוק את המילים temp בעמודה של referencing table :

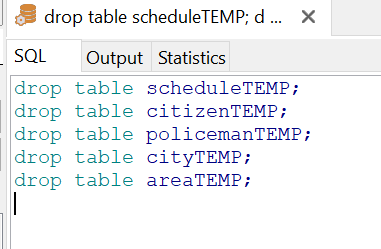


ביצענו את הפעולה הזאת בטבלאות הבאות:

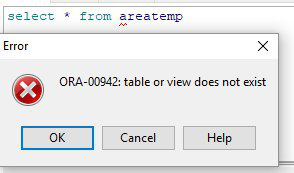
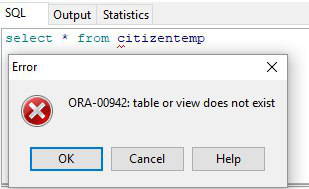
PolicemanContract, ResponseOn, Station.

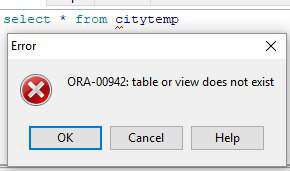
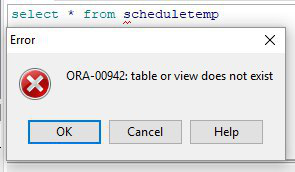
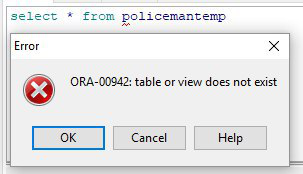
## 2.1.7 מחיקת טבלאות ה- temp

נמחק את הטבלאות שלקחנו מהם את הנתונים ע"י drop table.



הוכחה שהם אינם.





## 2.1.8 הקשר בין A ל- D

### **הקשר שעשינו:**

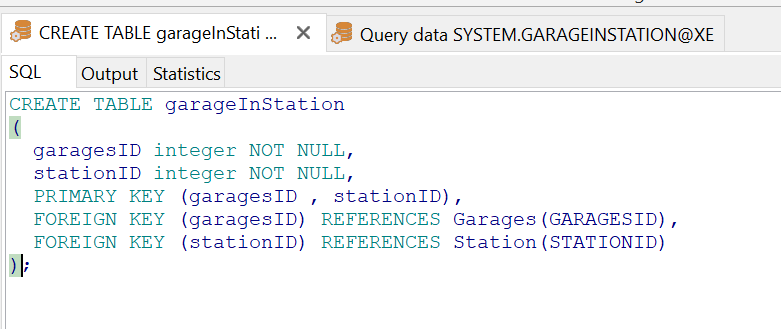
לכל תחנת משטרה (שמצאת באגף כוח אדם) יש קשר למוסך של רכבים (באגף רכב).

הקשר יהיה רבים לרבים, כלומר כל תחנה יכולה להיות קשורה לכמה מוסכים וכל מוסך לכמה תחנות משטרה. הקשר יראה בצורה אשר כתבנו לעיל שכל תחנה יכולה להיות קשורה לכמה מוסכים וכן להיפך, (יכול להיות כמה מוסכים באזור אחד וכן כמה תחנות משטרה באותו אזור).

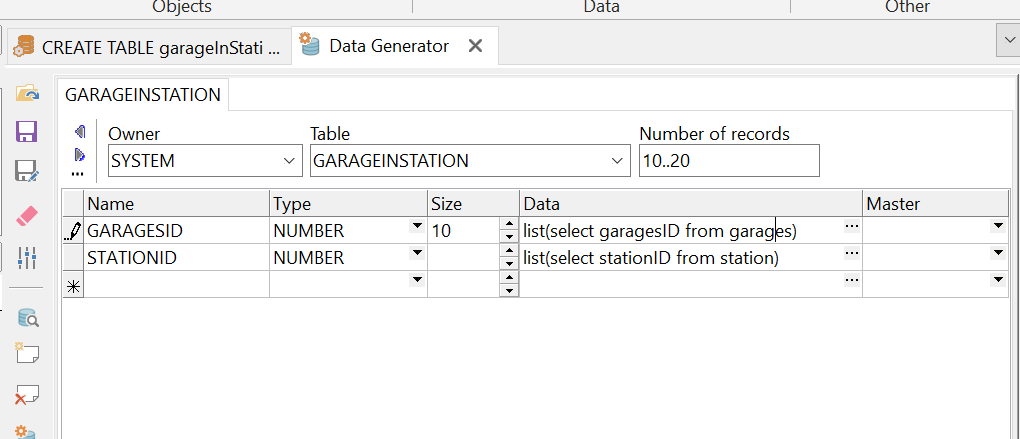
### **יצירת הטבלה**

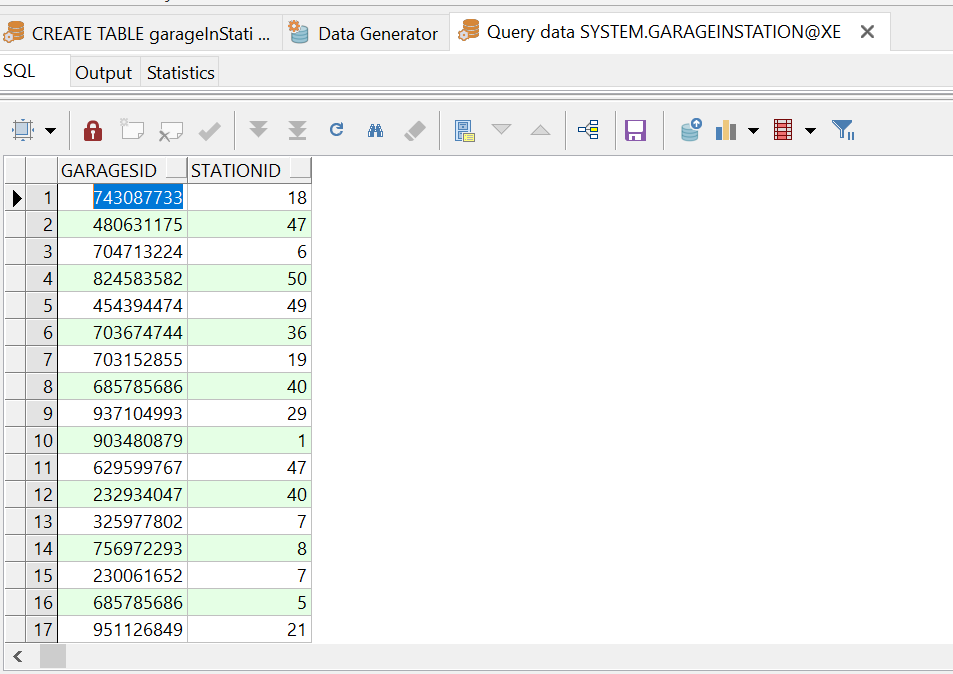
יצירת טבלה מקשרת בין מוסך לתחנה

ניצור טבלה אשר תכיל את ה PK של station וכן את ה PK של garages.



### **איכלוס הטבלה**

נראה את הנתונים שנכנסו ואשר מאחדים בין מחלקת כוח אדם למחלקת רכב.



# 3.0 שאילתות SQL

## 3.1 SQL Queries

### **שאילתא 1**

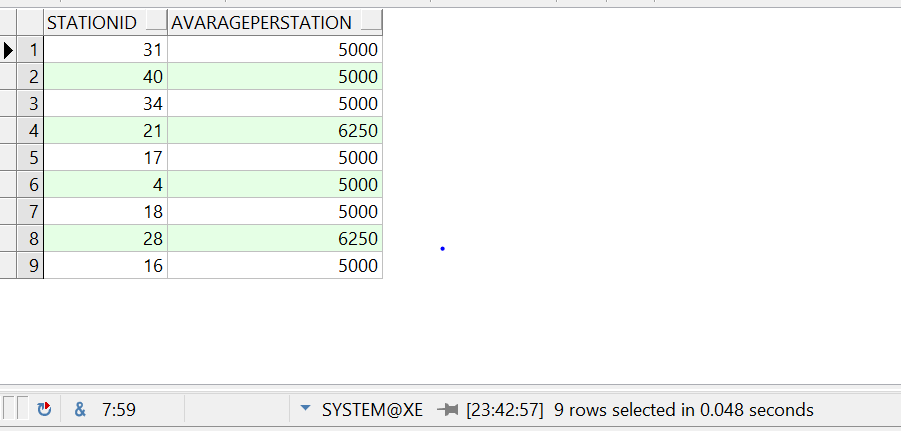
אגף כוח אדם רוצה לעודד את עובדי התחזוקה אך קיבל תקצוב מוגבל תוספת ל-9 עובדים בלבד, לכן רוצים לבדוק באילו תחנות קיימים פחות מ-10 עובדי תחזוקה והממוצע משכורת שלהם לתחנה קטן מ7000 ₪ לחודש.

select station.stationid ,avg(maintenanceagent.salary)

from station , maintenanceagent

where station.stationid = maintenanceagent.stationid

group by station.stationid

having count(\*)<10 and avg(maintenanceagent.salary)<7000

### **שאילתא 2**

מנהל אגף הרכב רוצה את שמות הספקים שיש איתם חוזים ליותר מ2 רכבים.

select suppliername

from suppliers

where supplierid in (select supplierid

from vehicle natural join vehiclecontract natural join suppliers

group by supplierid

 having count(\*)>2)

שאילתא חוצה אגפים

### **שאילתא 3**

רוצים לדעת איזה תחנות משטרה (שמות התחנות ומספריהן) משתמשות בשירותיו של המוסכים הכי עמוסים (שעברו בהם הכי הרבה רכבים) בשביל לנהל עומסים.

select distinct stationid, stationname

from station natural join

garageinstation natural join (

select garagesid

from vehicleingarages

group by garagesid

having count(\*) >= (select max(numOfVehiclesPerGarage)

from (select count(\*) as numOfVehiclesPerGarage

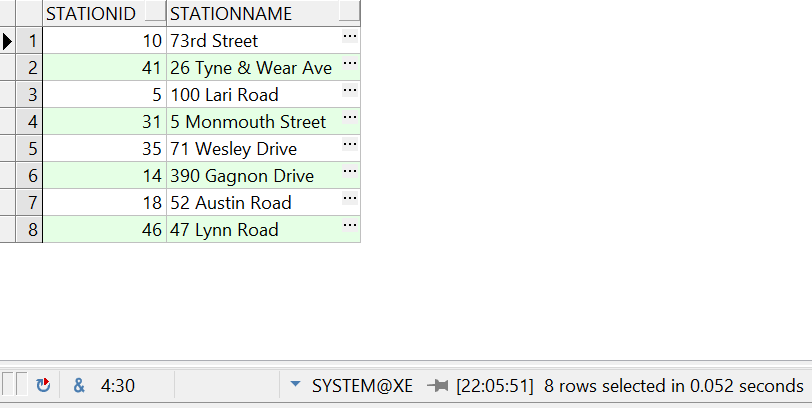
from vehicleingarages

group by garagesid

)

)

)



### **שאילתא 4**

מחלקת כחאדם רוצה להעלות את השכר של השוטרים המדורגים גבוה יחסית לממוצע של שאר השוטרים אך מרוויחים פחות מהממוצע של שאר השוטרים

SELECT PolicemanName

FROM Policeman P

WHERE P.RATING > (SELECT AVG(P.Rating)

FROM Policeman P)

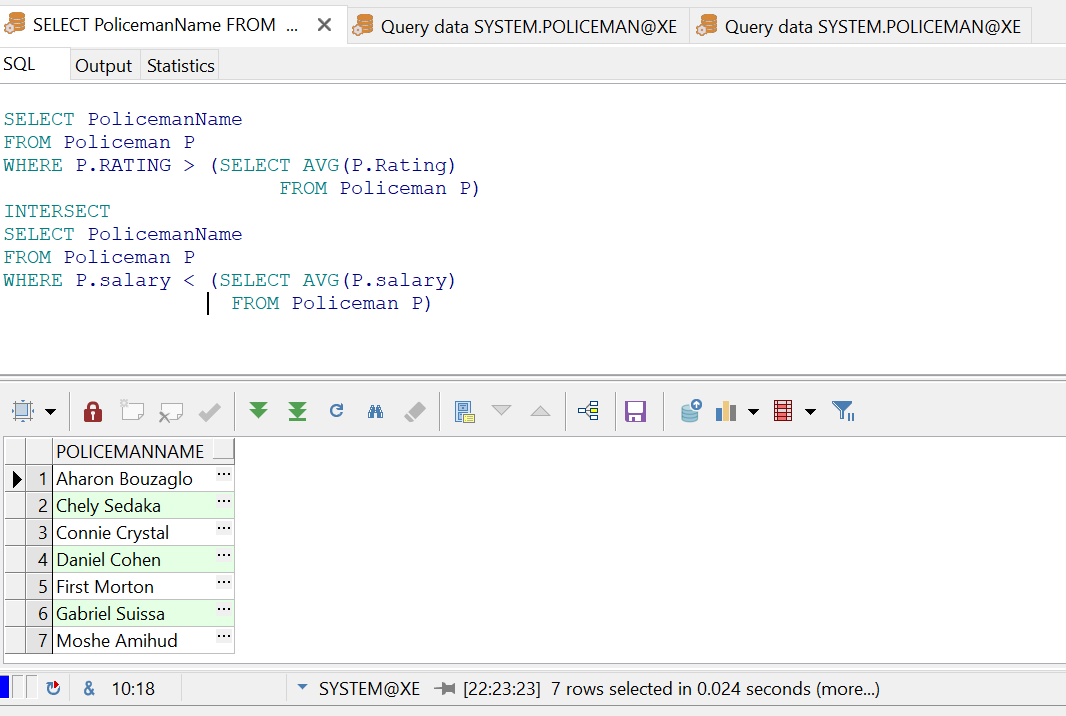
INTERSECT

SELECT PolicemanName

FROM Policeman P

WHERE P.salary < (SELECT AVG(P.salary)

FROM Policeman P)



### **שאילתא 5**

רוצים למצוא את שמות השוטרים ות.ז שמדורגים מעל דירוג 5 אך בפועל בזמן הנוכחי בינם לבין האזרחים יש פחות משתי פגישות מתוכננות!

SELECT policemanid , POLICEMANNAME

FROM Citizen c natural join schedule s natural join Policeman p Group by policemanid , POLICEMANNAME

having count(\*)<2

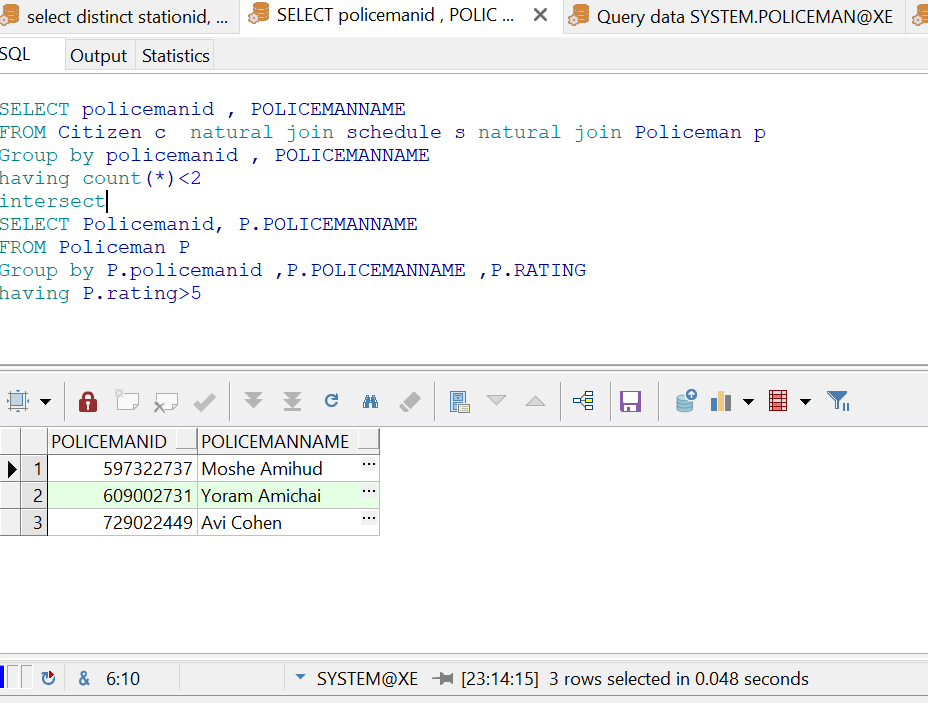
intersect

SELECT Policemanid, P.POLICEMANNAME

FROM Policeman P

Group by P.policemanid ,P.POLICEMANNAME ,P.RATING

having P.rating>5



### **שאילתא 6**

-כל התחנות ביהודה ושומרון

-מספר השוטרים בתחנה (לא כולל העובדי תחזוקה )

-ממוצע המשכורות בתחנה (לא כולל ה- extra salary)

create or replace View yehuda\_Veshomron\_Stations as

select stationID

from station natural join city natural join area

where areaid = ANY (select areaID

from area

where areaName = ' yehuda' or areaName = ' shomron');

*-------------------------------------------------------------------------------*

select stationID, stationName, count(\*) as Num\_Of\_Cops, round(avg(salary))

from policemanContract natural join station natural join policeman *--'station' to get its name.*

*--'policeman' to get the salary.*

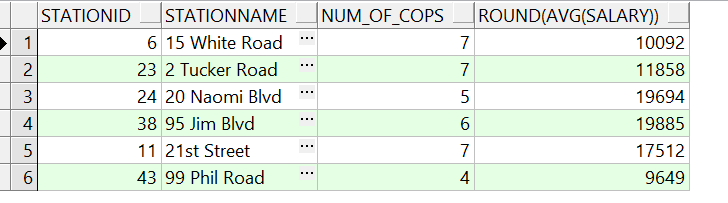
where stationID IN (select stationID from yehuda\_Veshomron\_Stations)

group by stationid, stationName

*הערה חשובה בטבלה policemanContract שמקשרת בין שוטרים לתחנה שלהם:*

*לכל שוטר שמופיע בטבלה יש בדיוק חוזה אחד לתחנה מסוימת. לכן כשמקבצים*

*לפי תחנה וסופרים כמה יש בכל קבוצה אז זה בדיוק מספר השוטרים בתחנה.*



### **שאילתא 7**

אנו רוצים להעלות את המשכורות של העובדים שמרוויחים הכי פחות מתוך קבוצת העובדים שהתחילו לעבוד לפני 2005 ומרוויחים פחות מ- 10,000₪ ולהציג את שמם ותעודת זהות בסדר עולה(לפי המשכורות)

select salary, agentid

from Maintenanceagent

where hireyear<2005 and salary <= ANY ( select salary

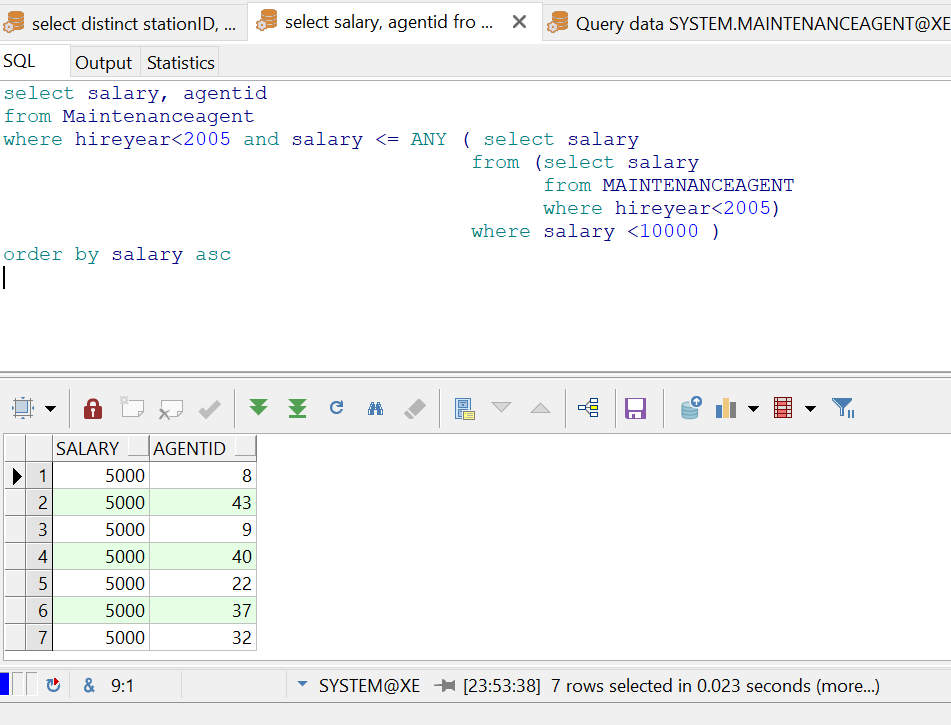
from (select salary

from MAINTENANCEAGENT

where hireyear<2005)

where salary <10000 )

order by salary asc



### **שאילתא 8**

מחלקת כח אדם מעוניינת לקבל את שמות כל העובדים במשטרה כולל העובדי תחזוקה. בנוסף כמה שנים הם עובדים במשטרה ומשכורתם.

select agentname as Worker\_Name,

(2019-hireyear) as years\_in\_the\_police

, salary

from maintenanceagent

union

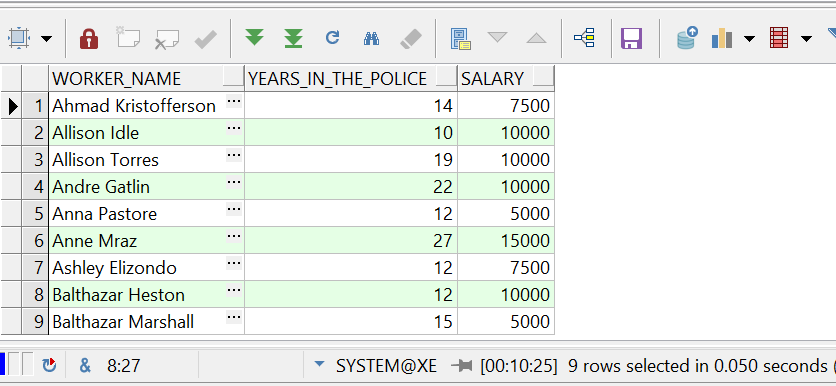
select policemanname as Worker\_Name,

(2019-hireyear) as years\_in\_the\_police

, salary

from policeman

where hireyear is not null



## 3.2 יצירת אינדקסים

**query 2**

CREATE INDEX IDX\_VEHICLE ON VEHICLE(CONTRACTID);

CREATE INDEX IDX\_VEHICLECONTRACT ON VEHICLECONTRACT(SUPPLIERID);

**query 3**

CREATE INDEX IDX\_GARAGEINSTATION ON GARAGEINSTATION(GARAGESID, STATIONID);

CREATE INDEX IDX\_VEHICLEINGARAGES ON VEHICLEINGARAGES(VEHICLEID, GARAGESID);

**query 5**

CREATE INDEX IDX\_CITIZEN ON Citizen(CityName);

CREATE INDEX IDX\_SCHEDULE ON SCHEDULE(CITIZENID,POLICEMANID);

**query 4, 5, 8**

CREATE INDEX IDX\_POLICEMAN ON POLICEMAN(AREAID);

**query 1, 7, 8**

CREATE INDEX IDX\_MAINTENANCEAGENT ON MAINTENANCEAGENT(STATIONID);

**query 1,3, 6**

CREATE INDEX IDX\_STATION ON STATION(CITYNAME);

We will make indexes on all the foreign keys, in case we'll need it for future queries

CREATE INDEX IDX\_CITY ON City(AreaID);

CREATE INDEX IDX\_POLICEMANCAR ON POLICEMANCAR(VEHICLEID);

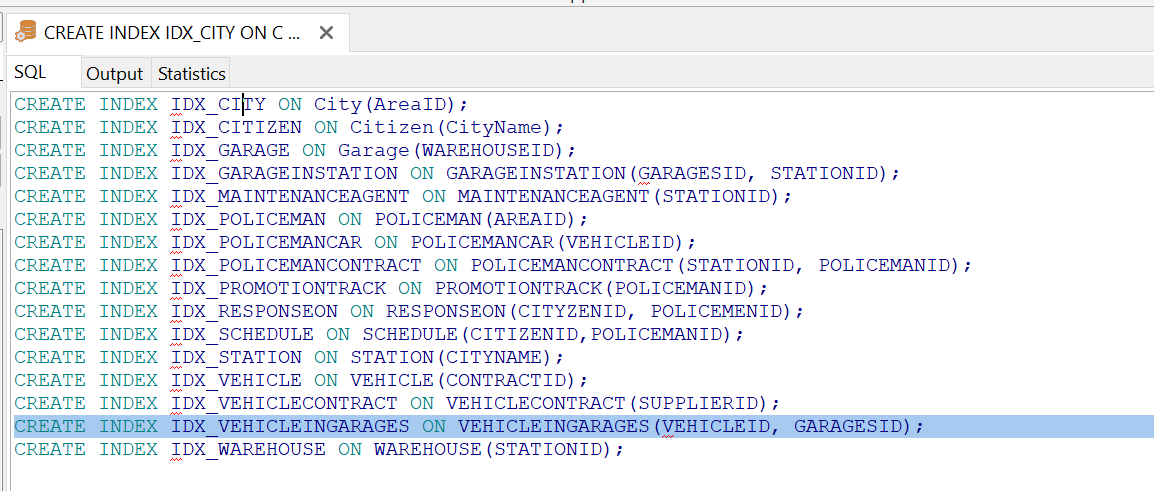
CREATE INDEX IDX\_POLICEMANCONTRACT ON POLICEMANCONTRACT(STATIONID, POLICMANID);

CREATE INDEX IDX\_PROMOTIONTRACK ON PROMOTIONTRACK(POLICEMANID);

CREATE INDEX IDX\_RESPONSEON ON RESPONSEON(CITYZENID, POLICEMENID);

CREATE INDEX IDX\_WAREHOUSE ON WAREHOUSE(STATIONID);

CREATE INDEX IDX\_GARAGE ON Garage(WAREHOUSEID);

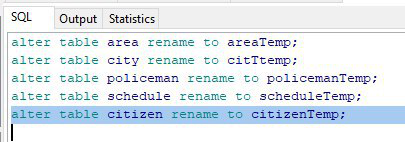
הרצה:

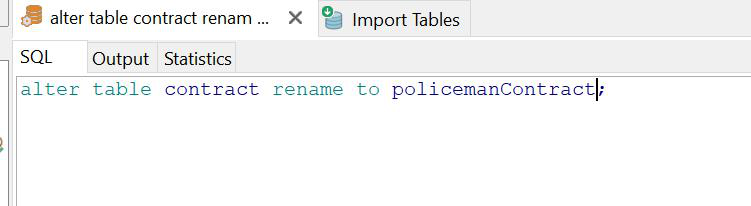
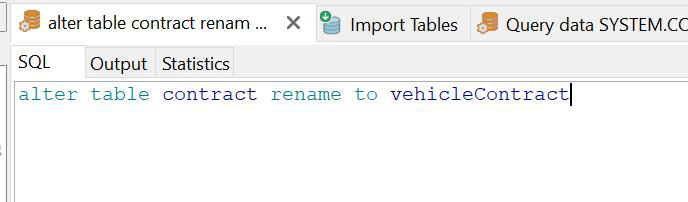
Comparison *Before / After* Indexing

|  |  |  |
| --- | --- | --- |
| Query # | Before Indexing | After Indexing |
| 1. | 0.048 sec | 0.025 sec |
| 2. | 0.016 sec | 0.011 sec |
| 3. | 0.052 sec | 0.029 sec |
| 4. | 0.024 sec | 0.024 sec |
| 5. | 0.048 sec | 0.029 sec |
| 6. | 0.0 sec | 0.0 sec |
| 7. | 0.023 sec | 0.019 sec |
| 8. | 0.050 sec | 0.022 sec |

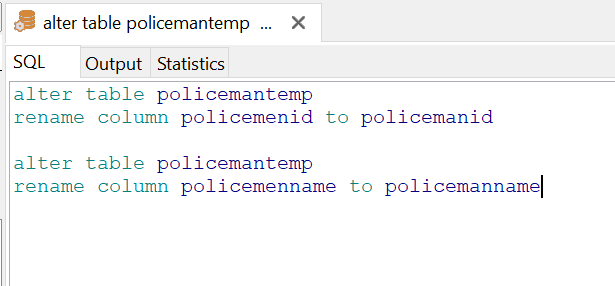
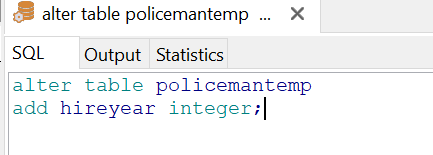
# 4.0 פקודות Alter

כל הפקודות נעשו בשלבים קודמים בפרויקט:

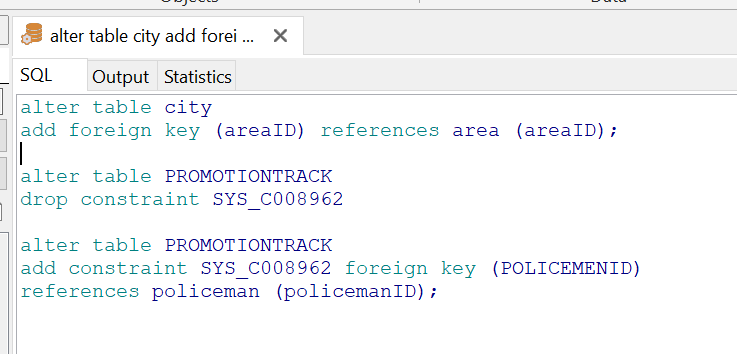
[עמוד 14 שינוי שם טבלאות](#_2.1.2_שינוי_השם)

[עמוד 15 שינוי שם טבלאות](#שינוי_שם_טבלאות)

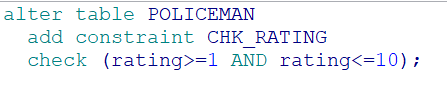
[עמוד 18 שינוי שם עמודות + הוספת עמודה](#_מעבר_מ-_policemamtemp)

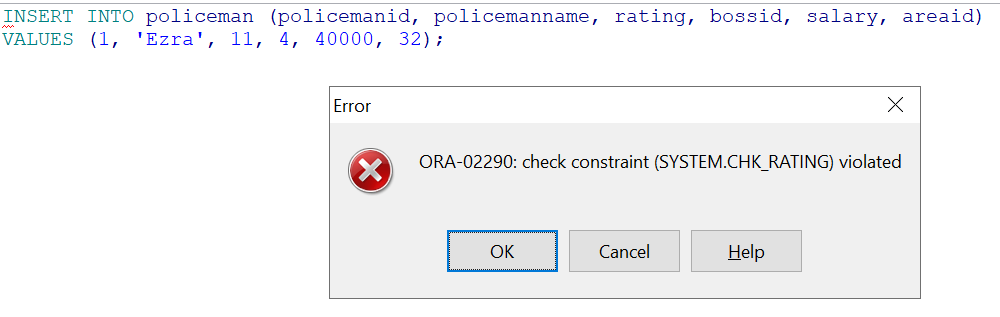
[עמוד 24 שינוי אילוצים והוספת מפתח זר](#_2.1.6_הפניית_FK)



הוספת אילוץ ב-Policeman על הטווח של rating



לכן אם ננסה להוסיף שוטר בדירוג 11 נקבל שגיאה



# 5.0 עדכון בסיס הנתונים – DML

## 5.1 פקודות Update

### **Update 1 - הדגמה של rollback & commit**

שאילתא שמציגה את שמות השוטרים, הדירוג שלהם, והשכר

select policemanname, rating, salary

from policeman

order by policemanname



*ב-update הראשון עשינו עדכון פשוט על מנת להדגים את כל נושא של ה-Rollback וה-Commit*

**שאילתת העדכון:**

מי שבדירוג מעל 4 – יעלה לו המשכורת ב1000 שקל.

update policeman

set salary = salary + 1000

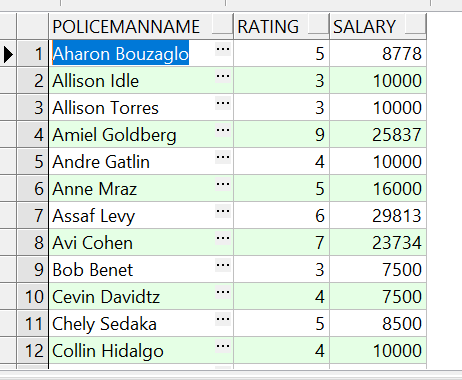
where rating > 4

**תוצאה לאחר השינוי:**

select policemanname, rating, salary

from policeman

order by policemanname



**Rollback:**

rollback;

select policemanname, rating, salary

from policeman

order by policemanname

*חזר להיות כמו מקודם:*



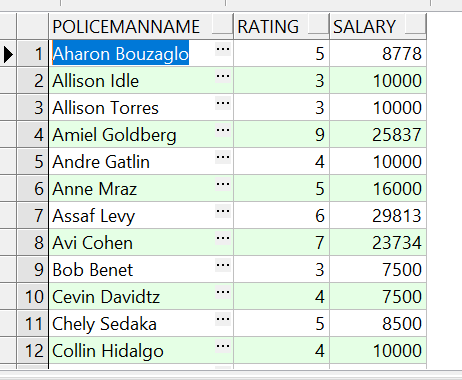
**שאילתת העדכון:**

update policeman

set salary = salary + 1000

where rating > 4

commit;



**Rollback:**

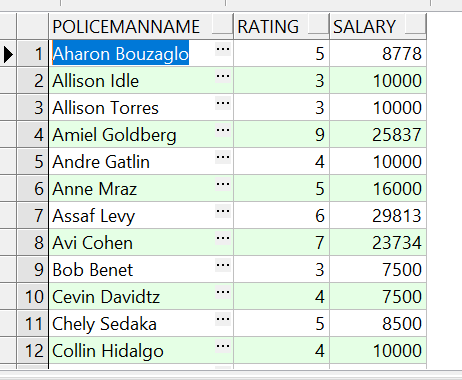
rollback;

select policemanname, rating, salary

from policeman

order by policemanname

*לא חזר אחורה בגלל שהתבצע Commit:*



### **Update 2 - חוצה אגפים**

שאילתא שמציגה את הטבלה שמקשרת בין האגפים – תחנת משטרה והמוסך שהיא משתמשת בו

select stationid, cityname, garagesid, areaid

from GARAGEINSTATION natural join station natural join garages



מכיוון שאין מוסכים באיזור 1 נרצה להעביר חלק מהמוסכים לאיזור זה.

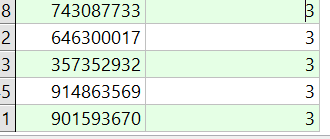
הוחלט שהמוסכים שמחזיקים ביותר מ2 רכבים יעברו לאיזור זה.

*אלה המוסכים שיש להם יותר מ-2 רכבים:*

select garagesID, count(\*) as numOfVehicles

from vehicleInGarages

group by garagesID



*נבצע את העדכון:*

update garages

set areaid = 1

where garagesID in (select garagesID

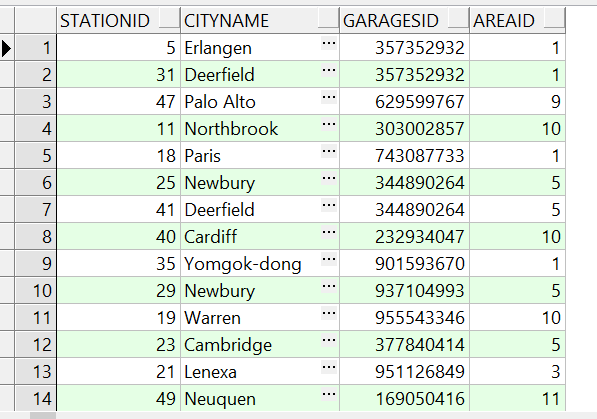
from vehicleInGarages

group by garagesID

having count(\*)> 2)

commit;

*אם נריץ את השאילתא למעלה שוב - ניתן לראות שהאיזור שלהם השתנה:*

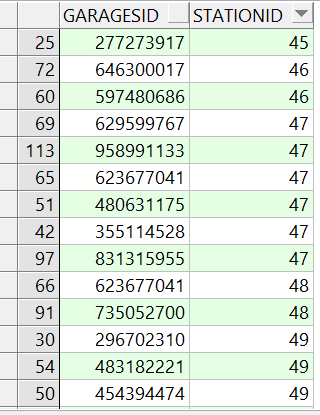


## 5.2 פקודות Delete

### **Delete 1 - חוצה אגפים**

שאילתא שמציגה את הטבלה שמקשרת בין האגפים – מוסכים והתחנות שמשתמשות בהם

select \* from GARAGEINSTATION



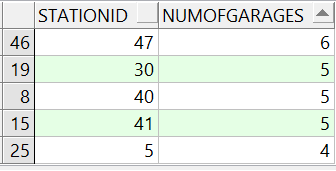
*החליטו שאם יש תחנה שמשתמשת ביותר מ-5 מוסכים – יבנו לה מוסך אחד גדול עם רכבים חדשים. לכן מה שרוצים לעשות זה למחוק את הקשר של תחנות אלה למוסכים הנוכחים שלהם.*

*יש תחנה אחת שמשתמשת ביותר מ-5 מוסכים – תחנה 47:*

select stationID, count(\*) as numOfGarages

from GARAGEINSTATION natural Join station

group by stationID



ולכן נמחק את הקשר לשישה המוסכים האלה:

delete

from garageInStation

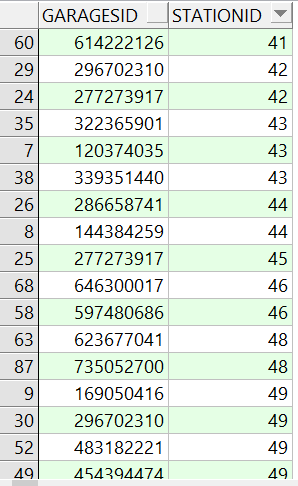
where stationID in (select stationID, count(\*) as numOfGarages

from GARAGEINSTATION natural Join station

group by stationID

having count(\*)>5 )

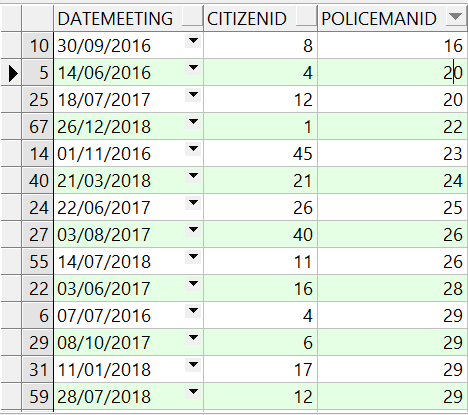
commit;



### **Delete 2**

שאילתא שמציגה את כל הפגישות בין השוטרים לאזרחים.

select \* from SCHEDULE t



החליטו לתת טיסה וביקור לאומן לכל השוטרים שמרוויחים מעל לממוצע והדירוג שלהם מעל לממוצע (למשל שוטר 26). לכן נבטל להם את כל הפגישות מהיומן:

delete

from schedule

where policemanID in( SELECT policemanID

FROM Policeman P

WHERE P.RATING > (SELECT AVG(P.Rating)

FROM Policeman P)

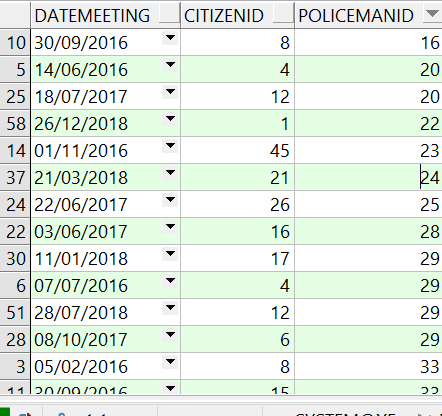
INTERSECT

SELECT policemanID

FROM Policeman P

WHERE P.salary > (SELECT AVG(P.salary)

FROM Policeman P) )



# 6.0 יצירת מבטים - Views

## View 1

הגדרת מבט בשם StationWithoutGarage אשר מחזיר את כל הפרטים של התחנות שלא משתמשות באף מוסך.

create or replace view StationsWithoutGarage as

(select stationID, stationname, cityname

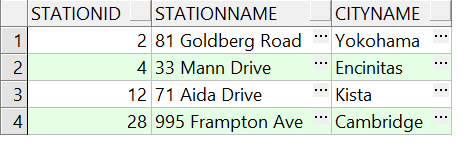
from station

minus

select distinct stationID, stationname , cityname

from garageInStation natural join garages natural join station);

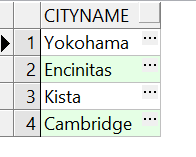
select \* from StationsWithoutGarage



**שאילתא המשתמשת במבט:**

select cityname

from StationsWithoutGarage



**:VIEW על ה- DML**

delete

from StationsWithoutGarage S

where S.cityname ='Yokohama

update StationsWithoutGarage

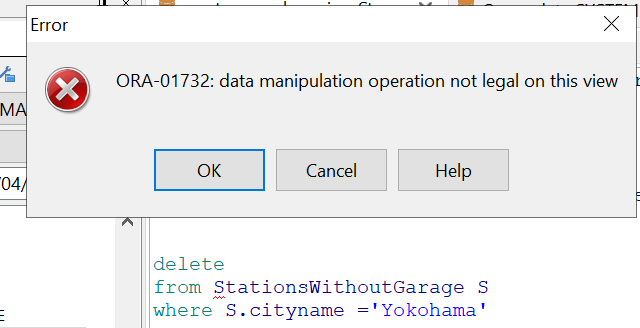
set cityname = 'Jerusalem'

where cityname ='Yokohama'

INSERT INTO StationsWithoutGarage (stationID, stationname, cityname)

VALUES (13, 'Skagen 21', 'Jerusalem');

**אותה שגיאה בכל פקודות ה-DML:**



*לא ניתן לשנות נתונים על מבטים שנוצרו ע"י איחוד טבלאות*

*כיוון שבטבלה שנוצרת יש שני מפתחות ראשיים – cityName ו- stationID*

## View 2

הגדרת מבט בשם policemenWithMeetings שמציג את הפרטים של כל השוטרים שיש להם לפחות פגישה אחת עם אזרחים.

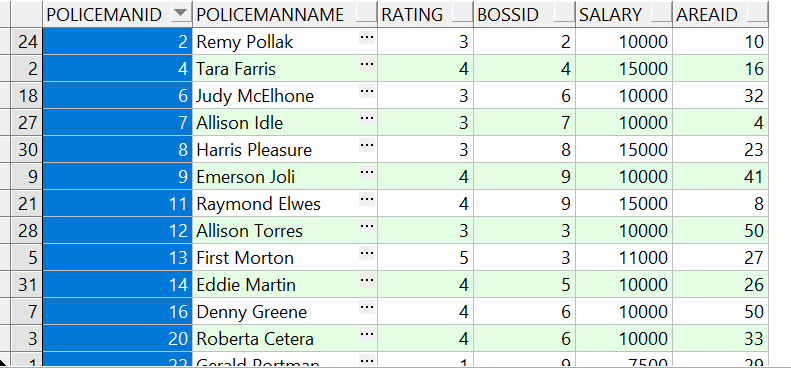
create or replace View policemenWithMeetings as

select policemanid, policemanname, rating, bossid, salary, areaid

from policeman

where policemanID in (select distinct policemanid from SCHEDULE);

select \* from policemenWithMeetings



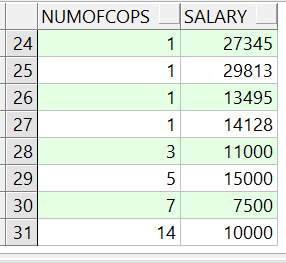
**שאילתא שמציגה כמה שוטרים מקבלים כל משכורת (מתוך המבט כמובן)**

select count(\*) as numOfCops , salary

from policemenWithMeetings

group by salary

order by numOfCops asc

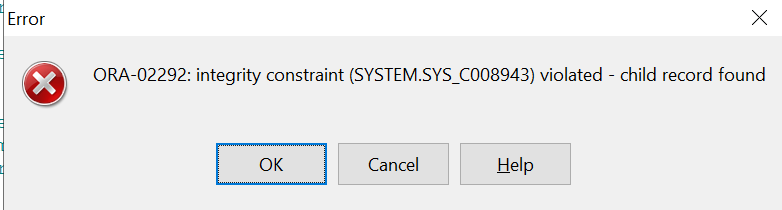


**:VIEW על ה- DML**

delete

from policemenWithMeetings P

where P.rating <= 4



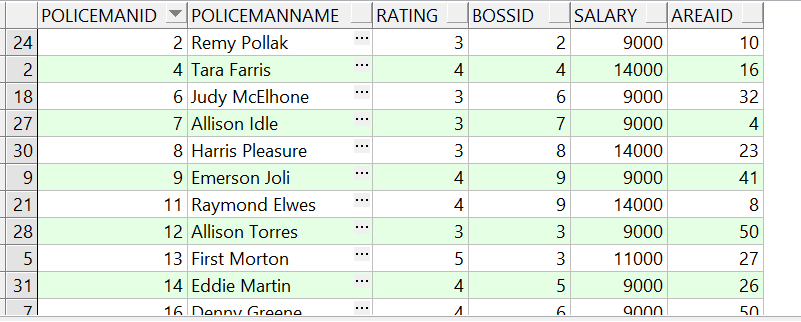
*אי אפשר למחוק שוטרים מסוימים מסיבה אחרת - מכיוון שלטבלאות אחרות יש מצביע זר לאותם שוטרים*

update policemenWithMeetings

set salary = salary-1000

where rating <= 4

commit;

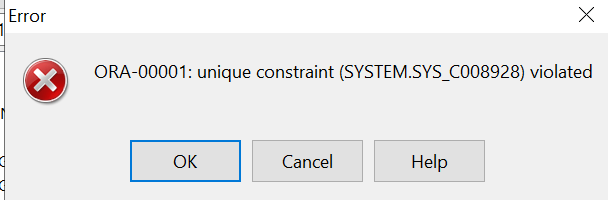


*עובד!! מכיוון שאין פה בעיה של שני מפתחות ראשיים (areaID – מפתח זר).*

*נציין שזה שינה גם את הטבלה של השוטרים ולא רק במבט.*

INSERT INTO policemenWithMeetings (policemanid, policemanname, rating, bossid, salary, areaid)

VALUES (1, 'Ezra', 10, 4, 40000, 32);



*קיים כבר שוטר עם מפתח זהה.*

*(ניסינו להוסיף שוטר עם מזהה 1 שלא קיים במבט, אך מכיוון שההוספה מתבצעת גם בטבלה של השוטרים אז האילוץ unique מופר)*

## View 3

הגדרת מבט בשם policemanList שמציג את כל השוטרים שנמצאים באזורים שיש בהם יותר מ-5 אזרחים.

create or replace View policemanList as

select \*

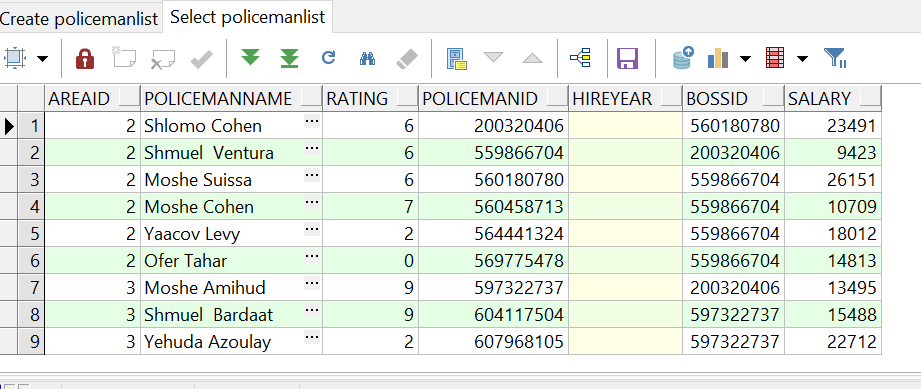
from (select areaid

from CITIZEN natural join city

group by areaid

having count(\*)>5) natural join policeman;

select \* from policemanList

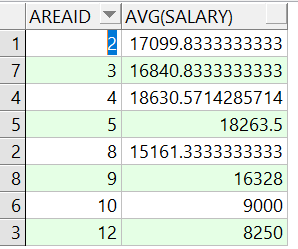


**שאילתא על המבט:**

select areaid, avg(salary)

from policemanList

group by areaid



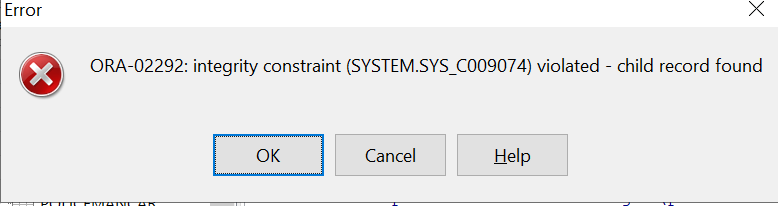
**:VIEW על ה- DML**

delete

from policemanList P

where areaid=2

commit;



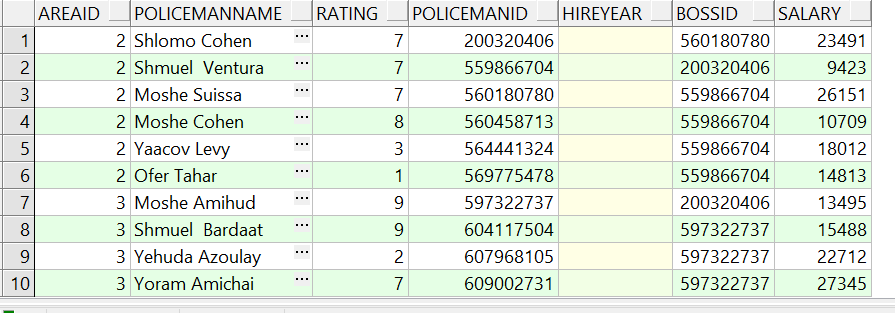
*אי אפשר למחוק שוטרים מסוימים מכיוון שלטבלאות אחרות יש מצביע זר לאותם שוטרים .*

update policemanList

set rating = rating + 1

where areaid=2

commit;



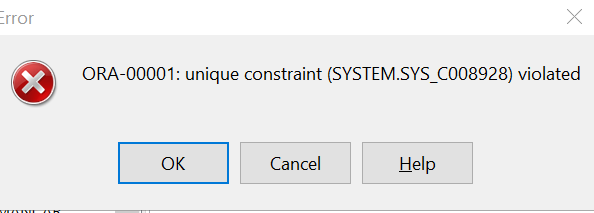
*עובד (אין בעיה של שתי מפתחות ראשיים). נציין שזה שינה גם את הטבלה של השוטרים ולא רק במבט.*

**ננסה להוסיף שוטר 38 למבט (לא נמצא שם כרגע)**

INSERT INTO policemanList (

areaid, policemanname, rating, policemanID, bossid, salary)

VALUES (38, 'Nils Newman',3, 1, 1, 15000);



מכיוון שהוא מוסיף אותו לטבלה של השוטרים (ולא רק למבט)

אז יש כבר שוטר עם מזהה 38 ולכן האילוץ Unique מופר.

# 7.0 תוכנית PL-SQL

## Program 1

בשלב האיחוד השתמשנו בתוכנית PL-SQL כדי למלא עמודה ריקה בתאריכים רנדומליים ([עמוד 22](#מילוי_תאריכים)). להלן התוכנית:

declare and implementaion

create or replace package myPackage is

*-- Author : OWNER*

*-- Created : 16/05/2019 19:57:36*

*-- Purpose :*

procedure insertRandomDates(dummy integer);

end myPackage;

-----------------------------------------------------------------

create or replace package body myPackage is

procedure insertRandomDates(dummy in integer)

is

cursor pm\_cur is select \* from schedule for update;

begin

for pm\_r in pm\_cur loop

update schedule

set dateMeeting = (SELECT TO\_DATE(

TRUNC(

DBMS\_RANDOM.VALUE(TO\_CHAR(DATE '2016-01-01','J')

,TO\_CHAR(DATE '2019-04-15','J')

)

),'J'

) FROM DUAL)

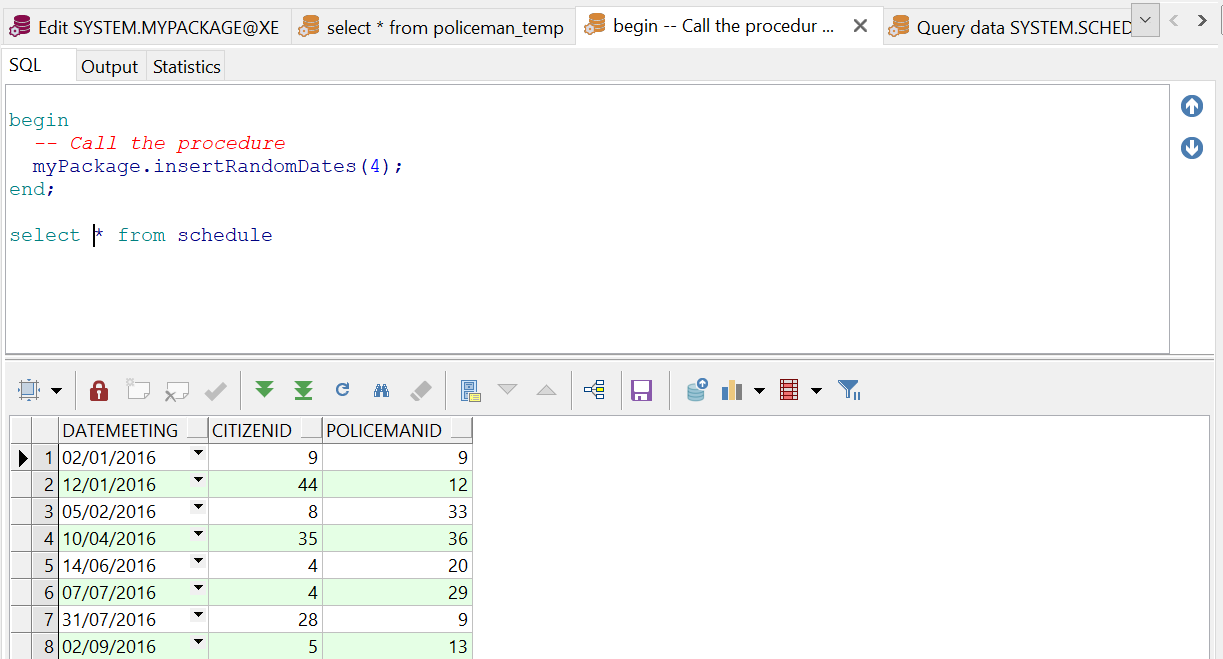
where current of pm\_cur;

end loop;

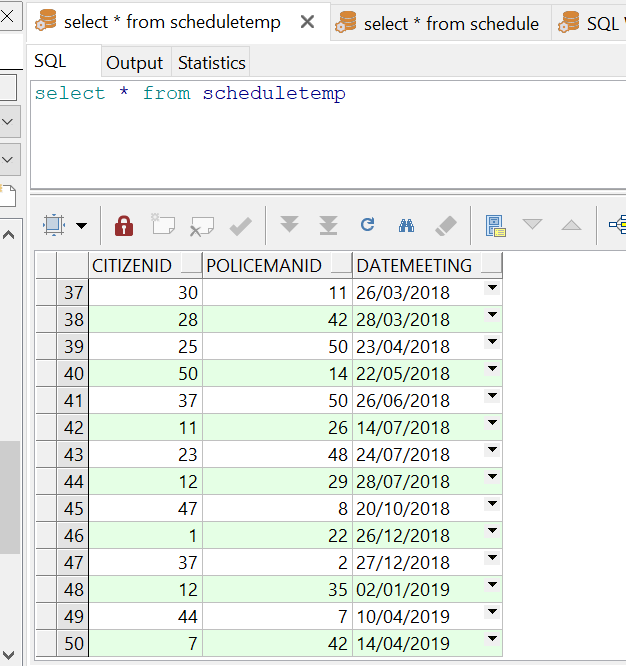
end;

end myPackage;

הרצת הפונקציה בתוכנית ראשית:



תוצאה:



## Program 2

נרשום תוכנית שתעדכן את המשכורות של השוטרים בדירוג מסוים. למשל נוסיף 1000 שקל לשוטרים בדירוג 7. כמו כן נוסיף (בהמשך) trigger שרושם בטבלה נפרדת את המשכורות בכל פעם שיש עדכון.

Declaration

create or replace noneditionable package myPackage is

*-- Author : OWNER*

*-- Created : 16/06/2019 19:57:36*

*--function <FunctionName>(<Parameter> <Datatype>) return <Datatype>;*

function ratingPoliceman(rating policeman.rating%type) return sys\_refcursor;

procedure updatePolicemanSalary(rating in number, addition in number);

end myPackage;

Implementation

create or replace noneditionable package body myPackage is

function ratingPoliceman(rating policeman.rating%type) return sys\_refcursor is

Result sys\_refcursor;

begin

open result for

select \* from policeman p

where p.rating >= rating;

return(Result);

end ratingPoliceman;

procedure updatePolicemanSalary(rating in number, addition in number) is

UpdatePoliceman sys\_refcursor;

oldSal policeman.salary%type;

newSal policeman.salary%type;

po policeman%rowtype;

begin

UpdatePoliceman := ratingPoliceman(rating);

LOOP

fetch UpdatePoliceman into po;

exit when UpdatePoliceman%notfound;

oldSal := po.salary;

newSal := oldSal+addition;

update policeman p

set p.salary = newSal

where p.policemanid = po.policemanid;

END LOOP;

end updatePolicemanSalary;

end myPackage;

בהתחלה נריץ רק את חלק התוכנית הבא:

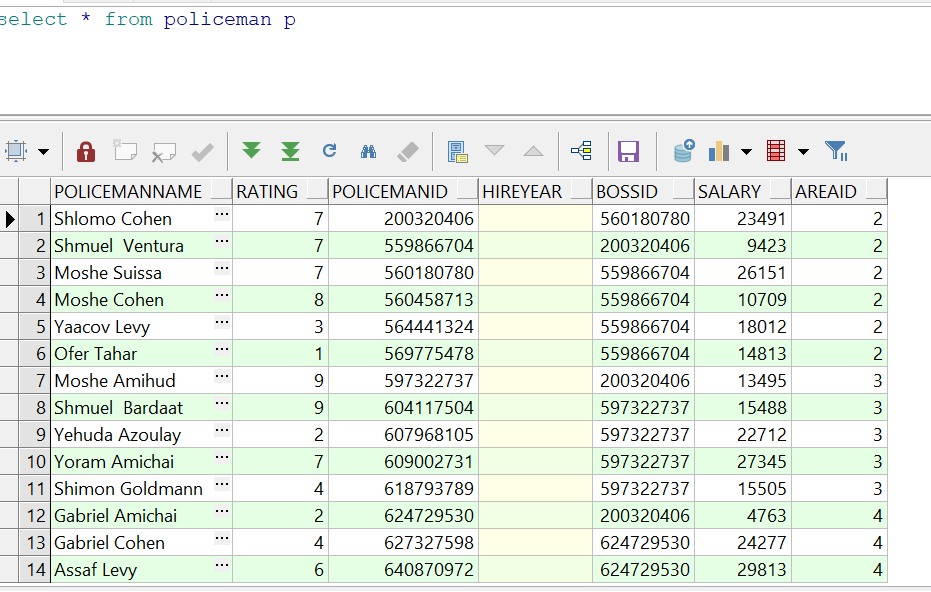
begin

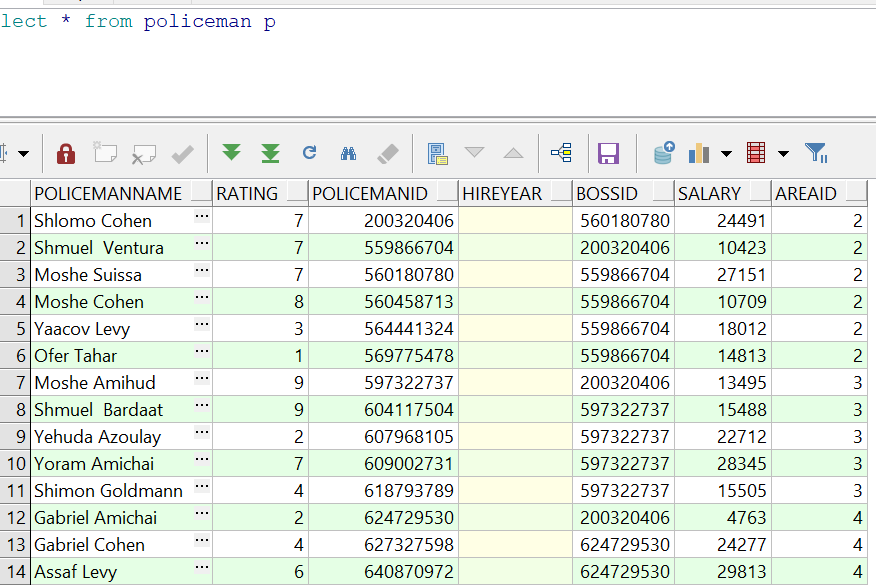
myPackage.updatePolicemanSalary(7,1000);

commit;

end;

לפני אחרי





בהמשך נוסיף לתוכנית מורכבות נוספת – נבדוק אם יש שוטרים בדירוג 7 שעדיין מקבלים מתחת ל-20,000 ₪ לאחר התוספת הראשונה. אם כן נוסיף להם עוד 1000 שקל למשכורת.

כמו כן נדפיס את הפרטים של השוטרים בדירוג 7 לאחר השינויים:

declare

myCursor sys\_refcursor;

oldSal policeman.salary%type;

newSal policeman.salary%type;

po policeman%rowtype;

begin

myPackage.updatePolicemanSalary(7,1000);

open myCursor for select \*

from policeman Policeman

where (Policeman.rating = 7) and (Policeman.Salary < 20000);

LOOP

fetch myCursor into po;

exit when myCursor%notfound;

oldSal := po.salary;

newSal := oldSal+1000;

update policeman p

set p.salary = newSal

where p.policemanid = po.policemanid;

END LOOP;

FOR myCursor IN (SELECT \* FROM policeman where rating=7)

LOOP

DBMS\_OUTPUT.PUT\_LINE('Policeman ID: ' || myCursor.policemanID);

DBMS\_OUTPUT.PUT\_LINE('Policeman Name: ' || myCursor.policemanName);

DBMS\_OUTPUT.PUT\_LINE('Salary: ' || myCursor.salary );

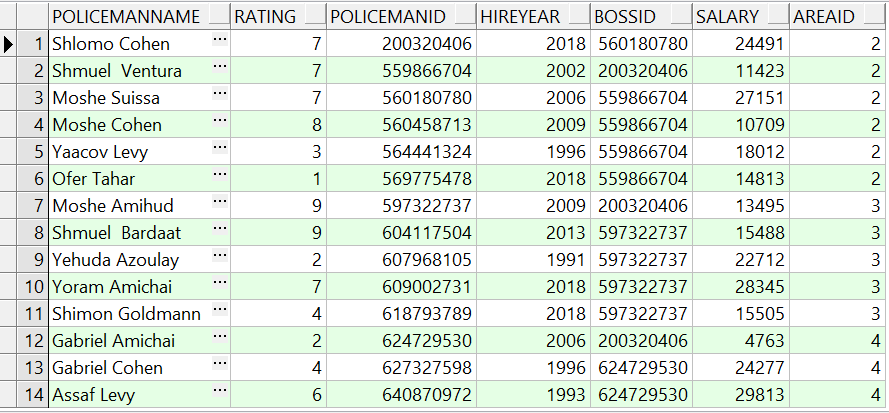
DBMS\_OUTPUT.PUT\_LINE('--------------------------------------------------');

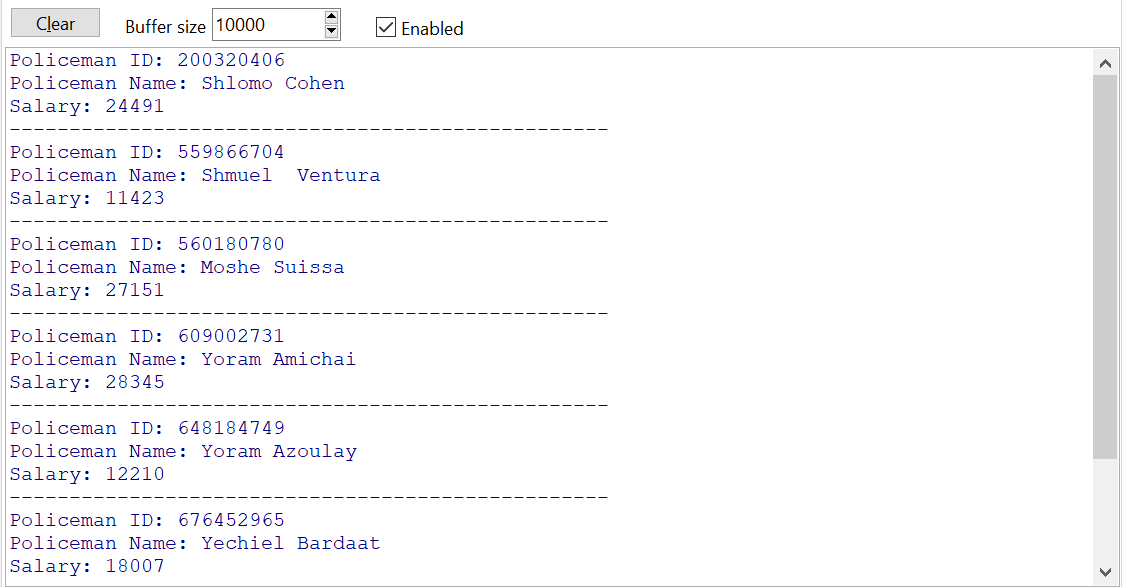
END LOOP;

commit;

end;

תוצאה:





### הוספת trigger

ניצור טבלה חדשה שתכיל מידע עבור עדכונים שבוצעו במשכורות של השוטרים.

create or replace trigger writeSalaryUpdates

after update on policeman

for each row

begin

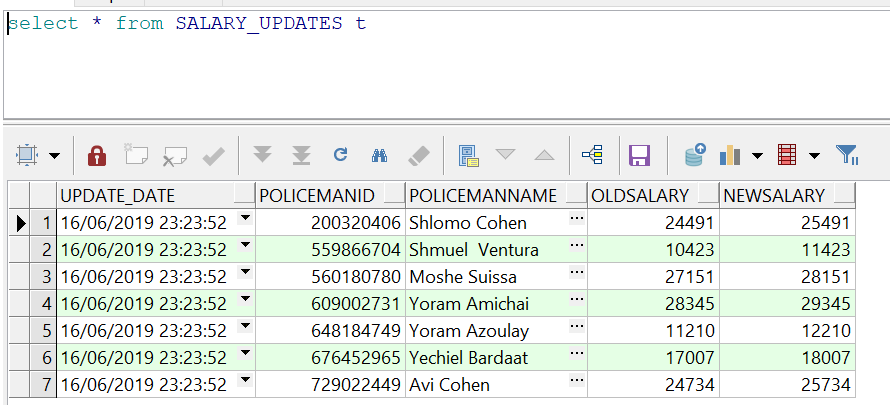
insert into Salary\_Updates

values(sysdate, :old.policemanID, :old.policemanName, :old.salary, :new.salary);

end writeSalaryUpdates;

להלן צילום מסך לאחר הרצת הפקודה פעם נוספת כאשר הטריגר מופעל:

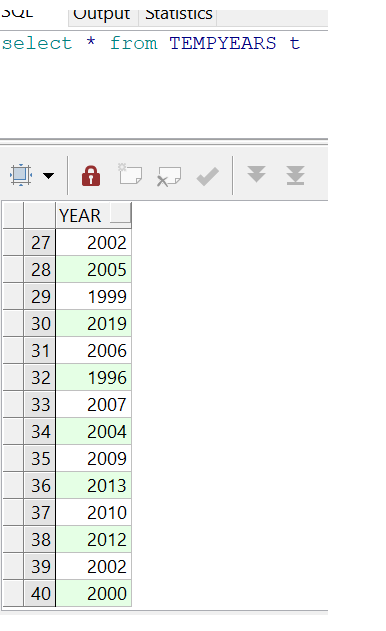
myPackage.updatePolicemanSalary(7,1000);



## Program 3

נמלא את העמודה - hireyear בטבלת הpoliceman לשוטרים שיש להם null (כזכור לחלק מהשוטרים העמודה ריקה מכיוון שבאגף משאבי אנוש לא היתה עמודה זו).

*ניצור טבלה זמנית של שנים ונמלא אותה עם ה-data generator:*



ועכשיו באמצעות תוכנית PL-SQL נמלא את החסר בעמודה hireyear, באמצעות

שליפה רנדומלית מהטבלת שנים שיצרנו:

declare

cursor pm\_cur is select \*

from policeman

where hireyear is null for update;

begin

for pm\_r in pm\_cur loop

update policeman

set hireyear = (select \* from (select T.year

from tempyears T

order by DBMS\_RANDOM.VALUE)

where rownum = 1

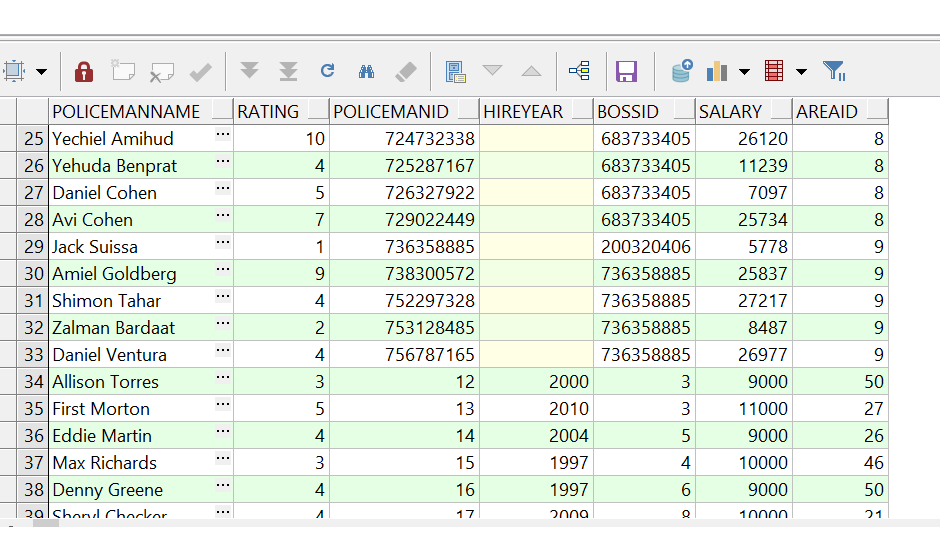
)

where current of pm\_cur;

end loop;

end;

**לפני:**



**אחרי:**

