מיני פרויקט בבסיסי נתונים

אגף התכנון של צה"ל



**מגישים:**

שלמה שפירא - 212280044

אלמוג זינגר - 206554941

**המרצה**:

יעקב ברזילאי

תוכן העיניינים:

שלב 1:   
 1.1 תיאור המערכת..................................................................................3

1.2 דיאגרמת ERD...................................................................................4

1.3 דיאגרמת DSD...................................................................................5  
 1.4 סקריפטים..........................................................................................8

1.5 טעינת מידע לבסיס הנתונים.................................................................12

1.6 גיבוי.................................................................................................20

שלב 2:   
 2.1 תיאור המערכת..................................................................................23

2.2 דיאגרמת ERD...................................................................................27

2.3 דיאגרמת DSD...................................................................................31  
 2.4 סקריפטים..........................................................................................35

2.5 טעינת מידע לבסיס הנתונים.................................................................39

**שלב 1**

* 1. **תיאור המערכת**

פיתוח מערכת ארגון שתסייע לאגף התכנון של צה"ל בניהול יעיל של תהליכי תכנון אסטרטגיים ופיתוח תוכניות מבצעיות לטוח ארוך.

באגף התכנון של צה"ל ישנם כמה יחידות שעוסקות כל אחת בפן אחר של המחקר והתכנון של הפעילויות המבצעיות והמודיעיניות של צה"ל.

בכל יחידה ינשם מספר חיילים שעובדים על מספר פרויקטים יחודיים לכל יחידה.

על המערכת לאפשר ניהול ושיבוץ הפרויקטים בהתאם צורך ובהתאם למשאבים ולחיילים הפנויים.  
כל פרויקט מורכב מהמשימות שלו ומאנשי הצוות שלו שמשויכים ליחידה לה יש משאבים המיוחדים ה.

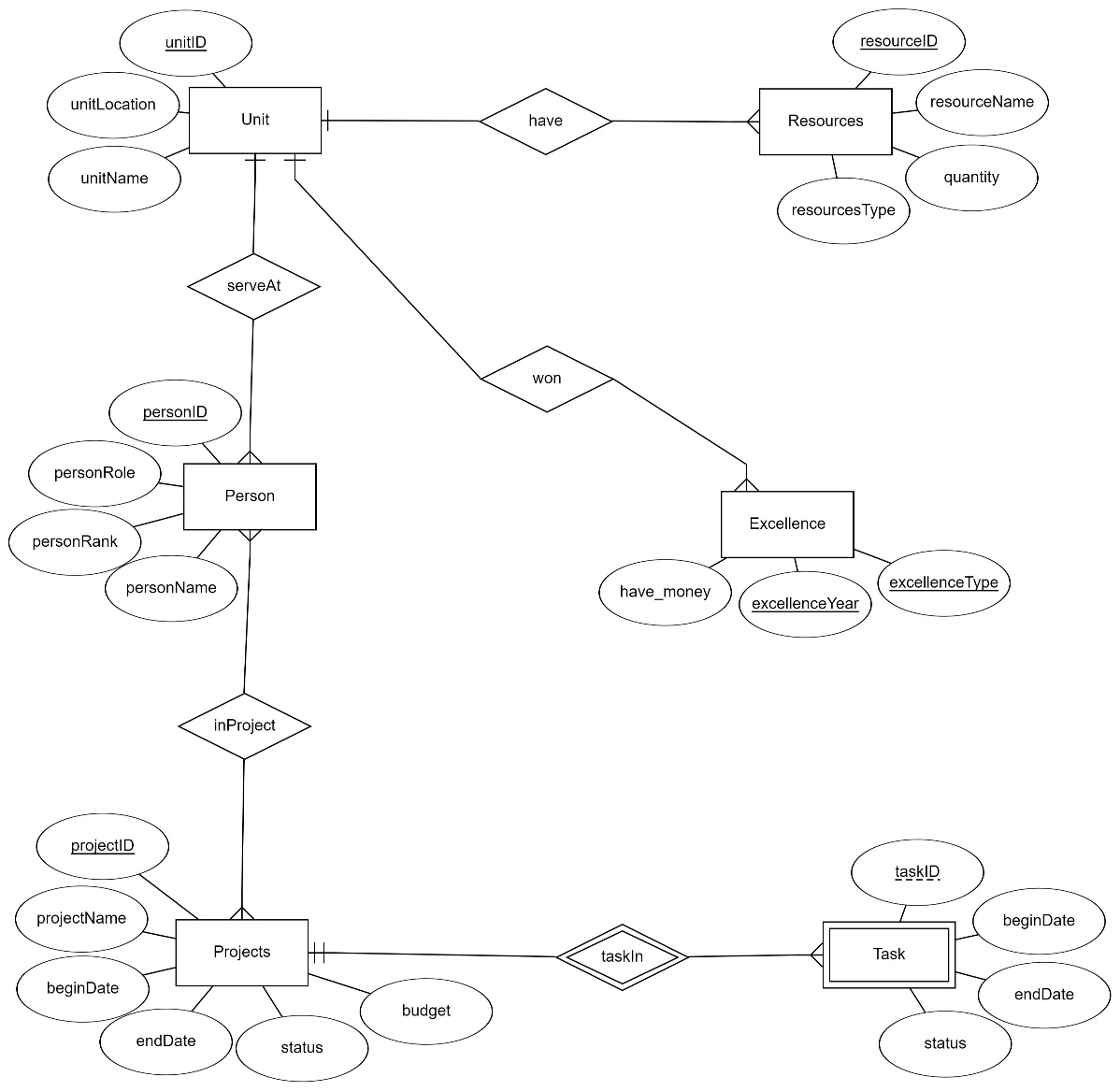
ניהול המערכת מתחלק ל3 חלקים עיקריים:

1. ניהול הפרויקטים : הפעלת פרויקט יוחלט לפי חשיבות , צוות מתאים ומשאבים פנויים.

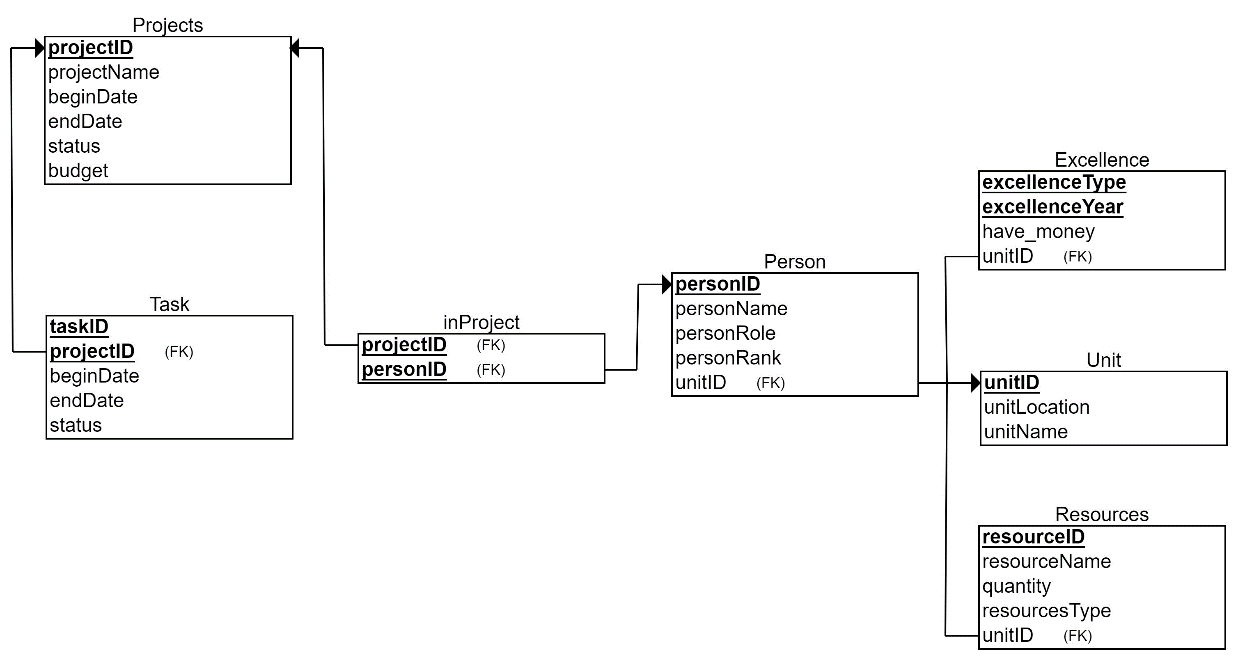
לכל פרויקט יוגדר זמן התחלה וזמן סיום על מנת לאפשר הגדרה של יחידה למספר פרוייקטים ללא התנגשות.

1. ניהול אנשי אגף המודיעין:   
   לכל יחידה יש כמה חיילים שמשובצים אליה. לכל יחידה ישנם מספר מוגדר של משאבים המוקצים לשימוש המחלקה , בין החיילים לפרויקט יש קשר היוצר את צוות הנאשים הממונים על הפרויקט.
2. ניהול אותות הערכה:   
   על מנת לאפשר תחרות ויצירתיות החליטו מפקדי האגף לייצר אותות הערכה שמיוצגים בטבלה .Excellence

**1.2 דיאגרמת ERD**



* 1. **דיאגרמת DSD**



**3NF:**

1. **טבלת Person:**

* **כל התכונות שאינן מפתח תלויות במפתח כולו:** personName, personRole, unitID תלויות ב-personID.

1. **טבלת Projects:**

* **כל התכונות שאינן מפתח תלויות במפתח כולו:** projectName, beginDate, endDate, status, budget, excellenceType, excellenceYear תלויות בprojectID.

1. **טבלת Task:**

* **כל התכונות שאינן מפתח תלויות במפתח כולו:** כל התכונות תלויות בtaskID.

1. **טבלת Excellence:**

* **כל התכונות שאינן מפתח תלויות במפתח כולו:** category, description, year תלויות ב-excellenceID.

1. **טבלת: Resources**

* **כל התכונות שאינן מפתח תלויות במפתח כולו:** resourceName, quantity, resourcesType תלויות ב-resourceID.

1. **טבלת Unit:**.

* **כל התכונות שאינן מפתח תלויות במפתח כולו:** unitName, unitLocation תלויות ב-unitID.

**הסכימה נמצאת ברמת נרמול 3NF מכיוון שלא קיימות תכונות מורכבות (1NF), אין תכונות**

**התלויות רק בחלק מהמפתח (2NF), ואין תכונות התלויות בתכונות אחרות שאינן חלק**

**המפתח (3NF).**

**1.3.1 תיאור הישויות**

* 1. **Person (אדם-חייל):** מתאפיין על ידי ID שזה המפתח הראשי שלו , שם , תפקיד , דרגה וID של היחידה בה משרת.
  2. **Project (פרוייקט):** מתאפיין על ידי ID (מפתח ראשי) שם הפרוייקט, תאריך התחלה , תאריך סיום משוער , מצב הפרוייקט, ותקציב שמוקצה לאותו פרוייקט.
  3. **Tesk (משימה) :** מתאפיין על ידי ID המשימה וID הפרוייקט שאליו הוא שייך. בנוסף קיים לו תאריך התחלת משימה ותאריך משוער של סיום המשימה. ישות חלשה של Project
  4. Unit **(יחידה) :** מתאפיינת על ידי הID מיקום ושם היחידה
  5. **Resource (משאב) :** מתאפיין על ידי ID שם המשאב , איכות , סוג המשאב וID היחידה שמובילה אותו.
  6. **Excellence(מצוינות) : אות הערכה** מתאפיין על ידי סוג האות והשנה שבה ניתן (שני השדות ביחד הם המפתח) והאם יש מענק כספי או לא (בוליאני F\T)

**1.3.2 תיאור הקשרים:**

* כל יחידה מחוברת (יש לה) למספר משאבים אך כל משאב מחובר ליחידה אחת ויחידה
* כל יחידה מחוברת (יכולה לקבל) למספר הצטיינויות אך הצטיינות מחוברת ליחידה אחת ויחידה
* כל אדם מחובר ליחידה אחת ויחידה מחוברת לכמה בני אדם
* כל פרויקט מחובר לכמה אנשי צוות וגם כל איש צוות יכול להיות מחובר לכמה פרויקטים
* משימה מחוברת לפרויקט בקשר של ישות חלשה.

**1.4 סקריפטים**

1.4.1 בניית טבלאות:

CREATE TABLE Projects

(

projectID INT NOT NULL,

projectName VARCHAR2(20) NOT NULL,

beginDate DATE NOT NULL,

endDate DATE NOT NULL,

status VARCHAR2(10) DEFAULT 'new' CHECK (status in ('new', 'start', 'middle', 'end' )),

budget FLOAT NOT NULL CHECK(budget >= 0),

PRIMARY KEY (projectID)

);

CREATE TABLE Unit

(

unitID INT NOT NULL,

unitLocation VARCHAR2(20) NOT NULL,

unitName VARCHAR2(20) NOT NULL,

PRIMARY KEY (unitID)

);

CREATE TABLE Resources

(

resourceID INT NOT NULL,

resourceName VARCHAR2(20) NOT NULL,

quantity INT NOT NULL,

resourcesType VARCHAR2(10) NOT NULL,

unitID INT NOT NULL,

PRIMARY KEY (resourceID),

FOREIGN KEY (unitID) REFERENCES Unit(unitID)

);

CREATE TABLE Task

(

taskID INT NOT NULL,

projectID INT NOT NULL,

endDate DATE NOT NULL,

beginDate DATE NOT NULL,

status VARCHAR2(10) DEFAULT 'new' CHECK (status in ('new', 'start', 'middle', 'end' )),

PRIMARY KEY (taskID, projectID),

FOREIGN KEY (projectID) REFERENCES Projects(projectID)

);

CREATE TABLE Excellence

(

excellenceType VARCHAR2(20) NOT NULL,

excellenceYear NUMBER(4) NOT NULL,

have\_money CHAR NOT NULL CHECK(have\_money in ('T', 'F')),

unitID INT NOT NULL,

PRIMARY KEY (excellenceType, excellenceYear),

FOREIGN KEY (unitID) REFERENCES Unit(unitID)

);

CREATE TABLE Person

(

personID INT NOT NULL,

personName VARCHAR2(20) NOT NULL,

personRole VARCHAR2(20) NOT NULL,

personRank VARCHAR2(20) NOT NULL,

unitID INT NOT NULL,

PRIMARY KEY (personID),

FOREIGN KEY (unitID) REFERENCES Unit(unitID)

);

CREATE TABLE inProject

(

projectID INT NOT NULL,

personID INT NOT NULL,

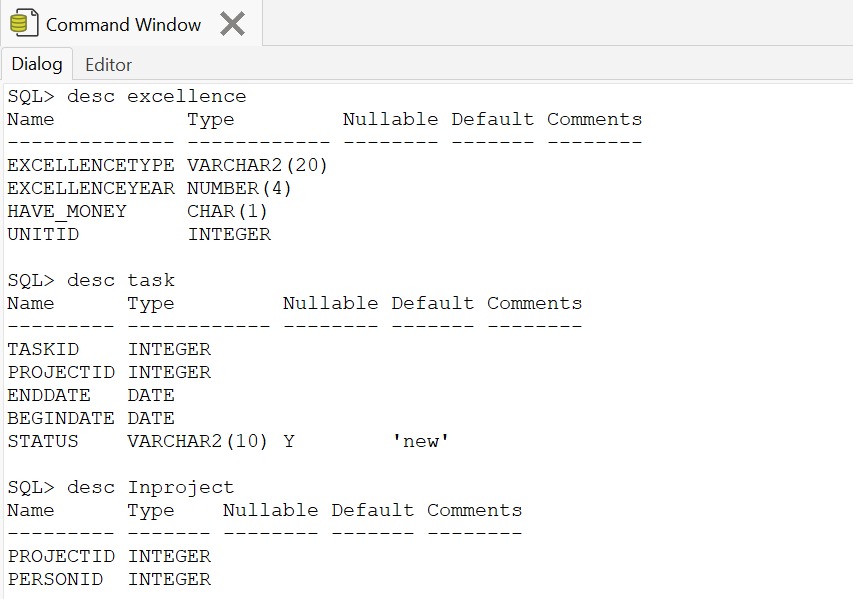
PRIMARY KEY (projectID, personID),

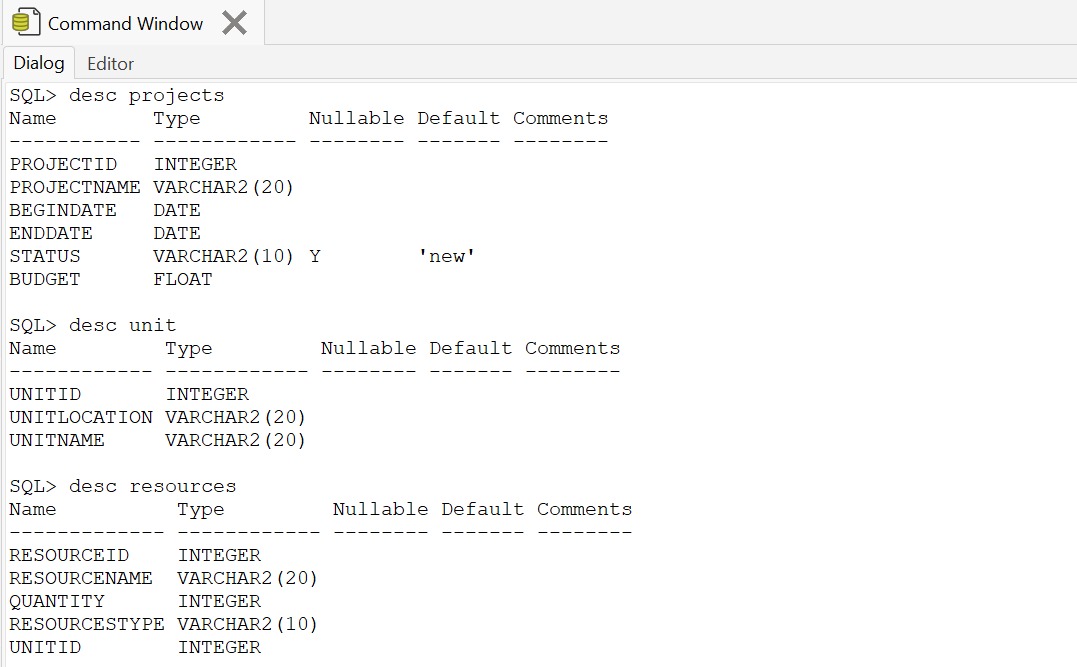
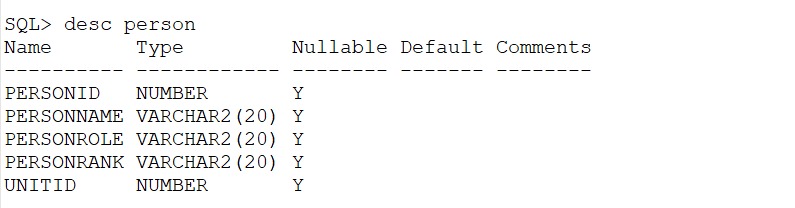
FOREIGN KEY (projectID) REFERENCES Projects(projectID),

FOREIGN KEY (personID) REFERENCES Person(personID)

);

1.4.1 פקודות desc



1.4.3 מחיקת הטבלאות

DROP TABLE inProject;

DROP TABLE Task;

DROP TABLE Projects;

DROP TABLE Person;

DROP TABLE Excellence;

DROP TABLE Resources;

DROP TABLE Unit;

1.4.4 בחירה מתוך הטבלאות

SELECT \* FROM inProject;

SELECT \* FROM Task;

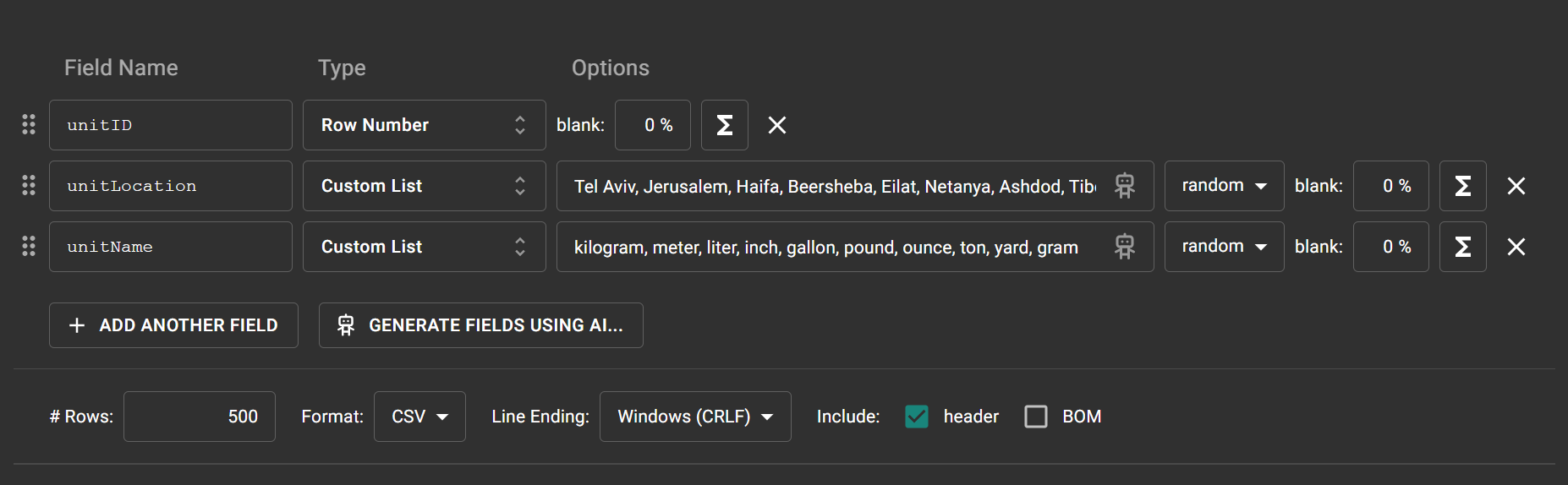
SELECT \* FROM Projects;

SELECT \* FROM Person;

SELECT \* FROM Excellence;

SELECT \* FROM Resources;

SELECT \* FROM Unit;

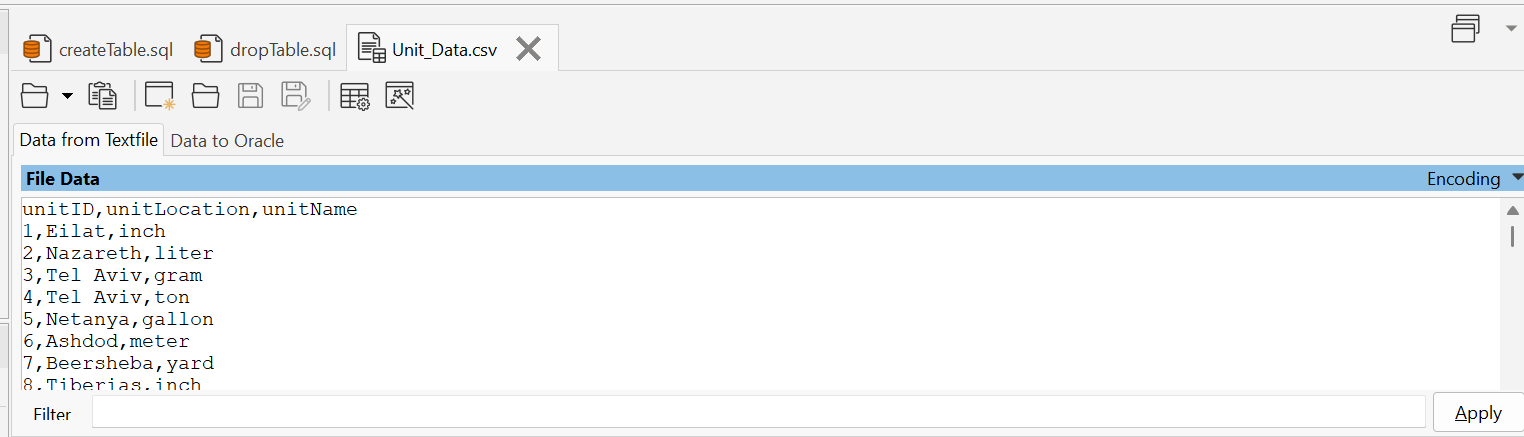
**1.5. טעינת מידע לבסיס הנתונים:**

הערה: השתמשנו בעצם ב-4 דרכים כי ג'ינרטנו נתונים עם mockaroo, אלא שבמקום ליצור ישר שאילתות, יצרנו מזה קובץ CVS שאותו ייבאנו לפרויקט.  
***הכנסה עם מחולל נתונים וקובץ cvs:***  
  
**הטבלה UNIT:**

תמונה שמכילה טקסט, צילום מסך, תוכנה, מספר

התיאור נוצר באופן אוטומטינייבא את זה ל-plsql בעזרת importText.

נבחר את הקובץ המבוקש, נבחר את הטבלה שאליה רוצים להכניס את הנתונים ואז נלחץ import.

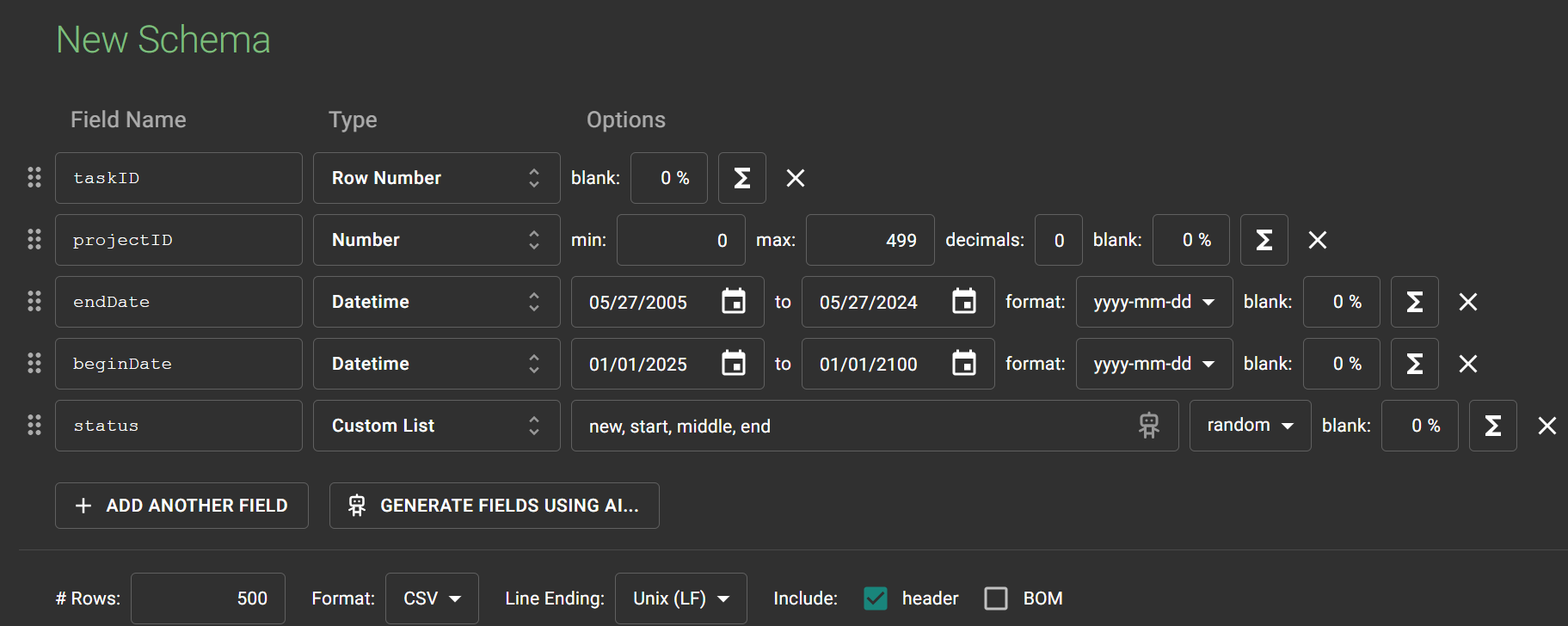


לאחר ההוספה, זאת התוצאה של פקודת ה-select:

תמונה שמכילה טקסט, צילום מסך, תוכנה, מספר

התיאור נוצר באופן אוטומטי

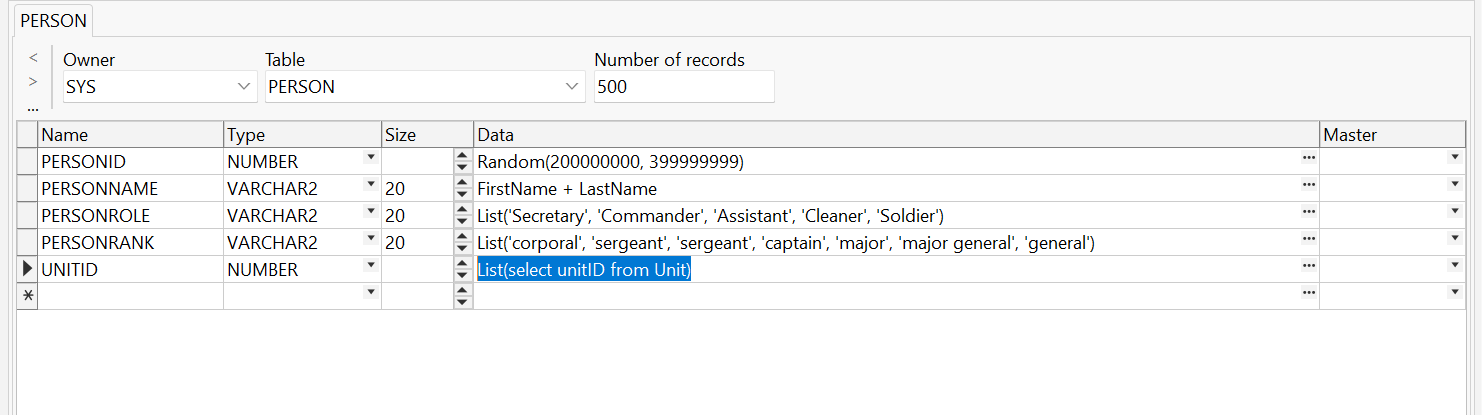
**הטבלה Task:**

התהליך זהה  


תמונה שמכילה טקסט, צילום מסך, תוכנה, מספר

התיאור נוצר באופן אוטומטי  
  
  
לאחר ההוספה:   
תמונה שמכילה טקסט, צילום מסך, תוכנה, תצוגה

התיאור נוצר באופן אוטומטי

**דרך שנייה : דרך הgenerate בplsql:  
הטבלה person:**

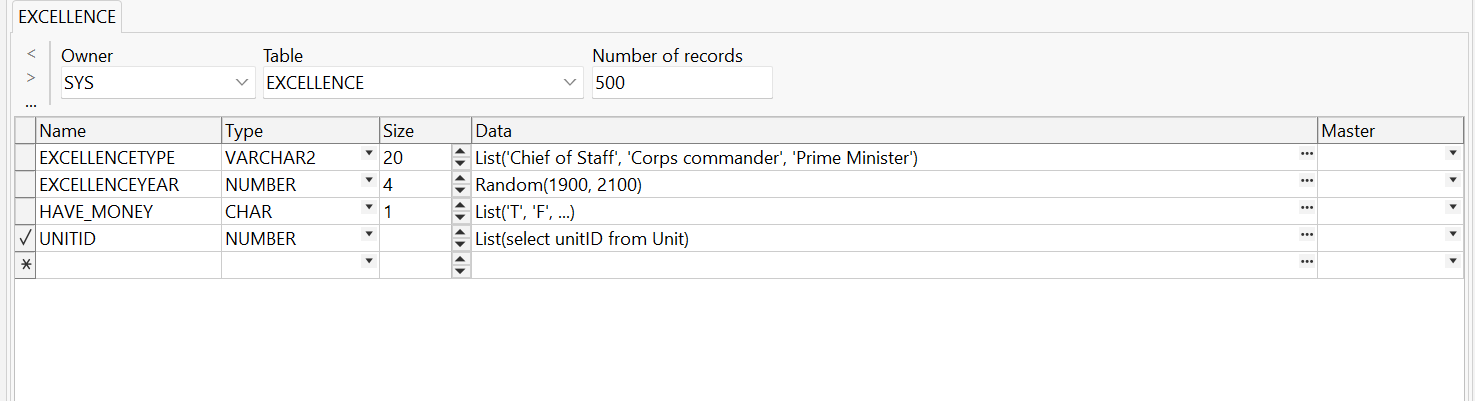
נשים לב שאת הערך unitID שהוא מפתח זר, נבחר מתוך נתוני הטבלה UNIT.

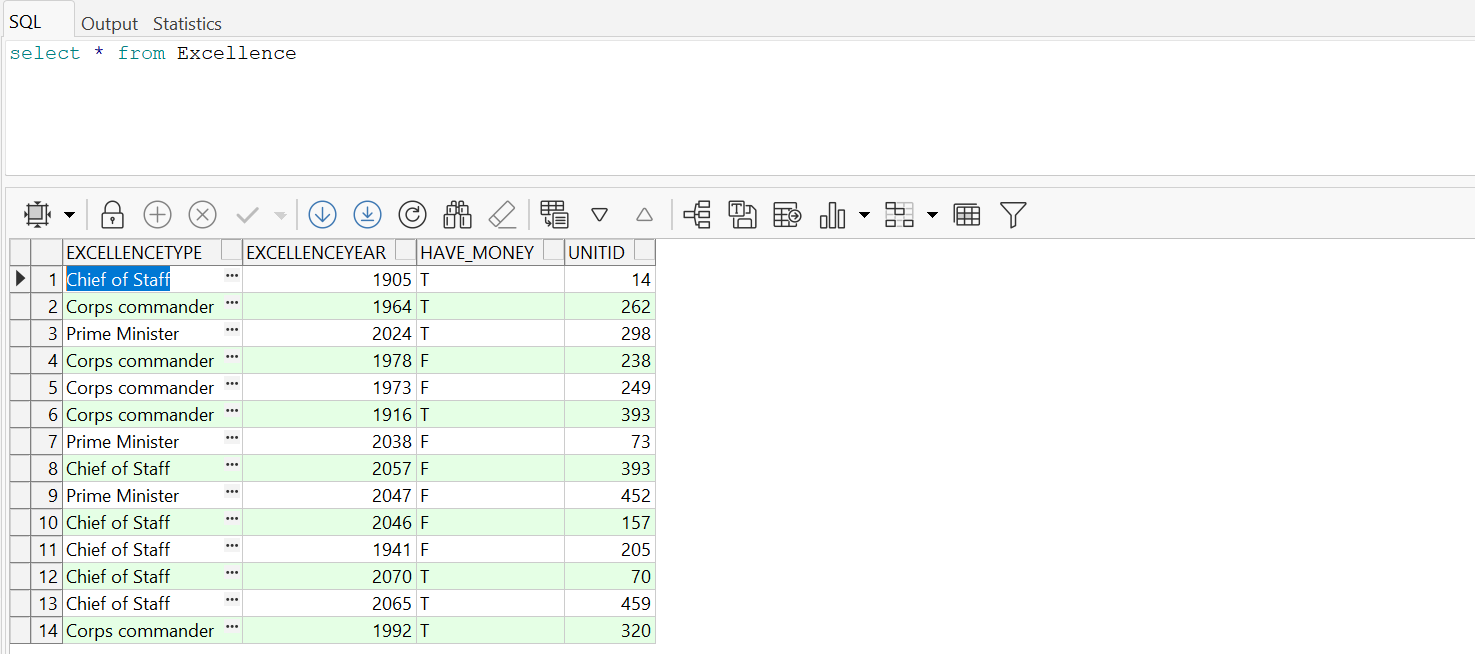
לאחר ג'ינרוט הנתונים:

תמונה שמכילה טקסט, חשמל, צילום מסך, מספר

התיאור נוצר באופן אוטומטי

**אותה דרך – הטבלה excellence:**

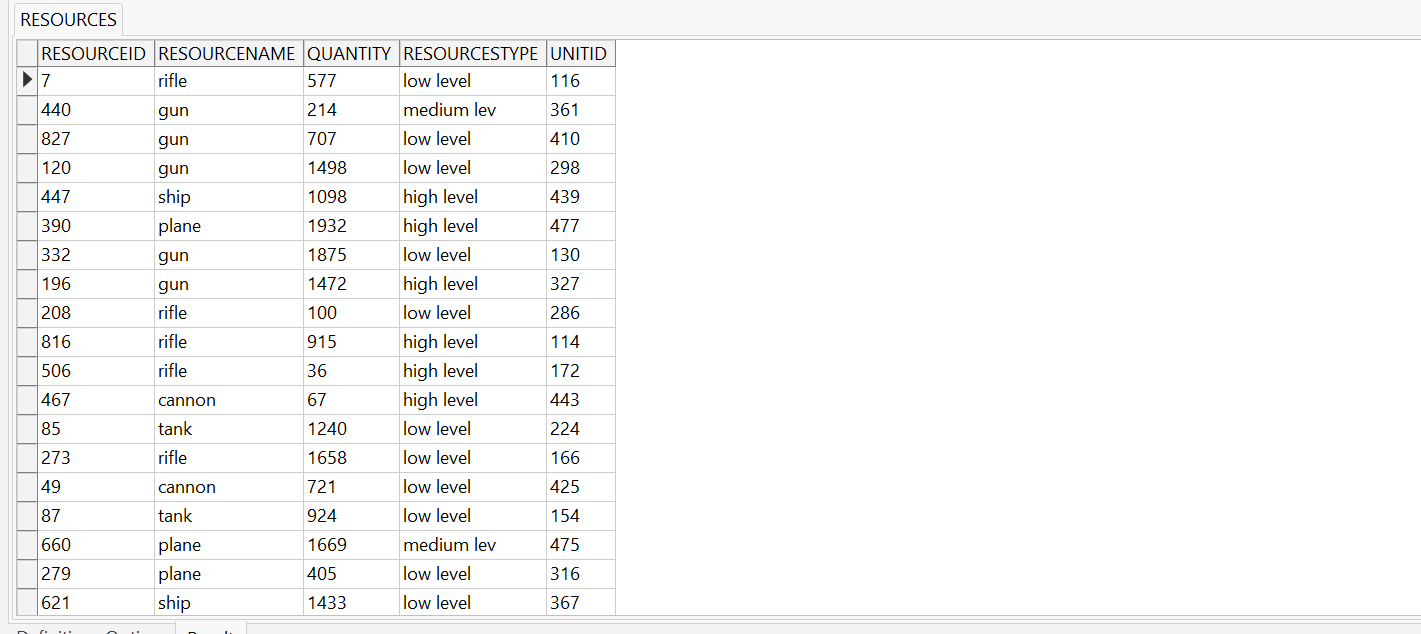




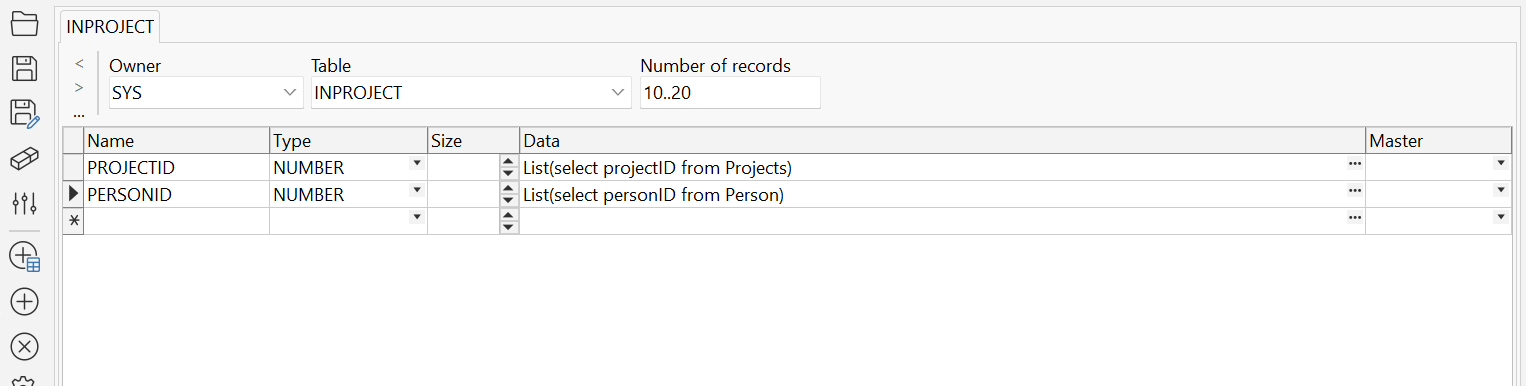
**הטבלה Resurce::**

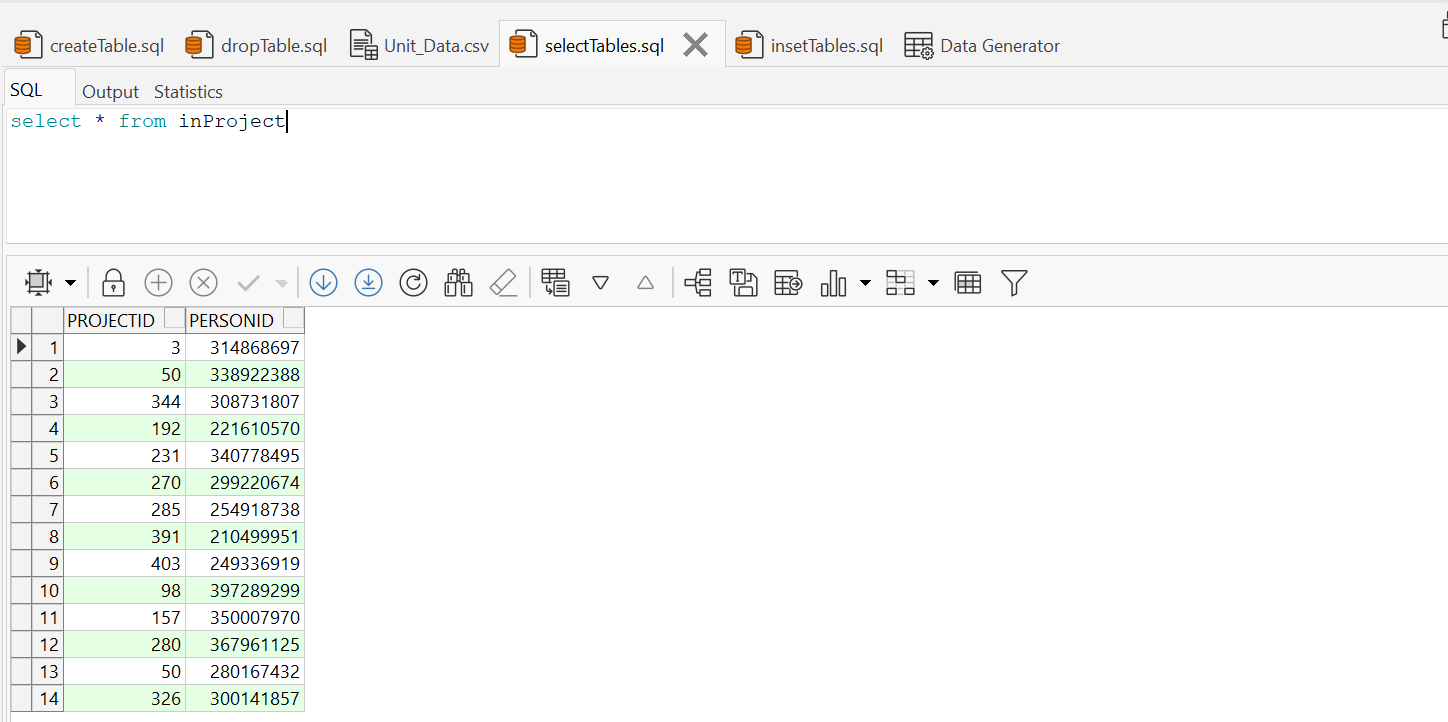
תמונה שמכילה טקסט, צילום מסך, קו, גופן

התיאור נוצר באופן אוטומטי



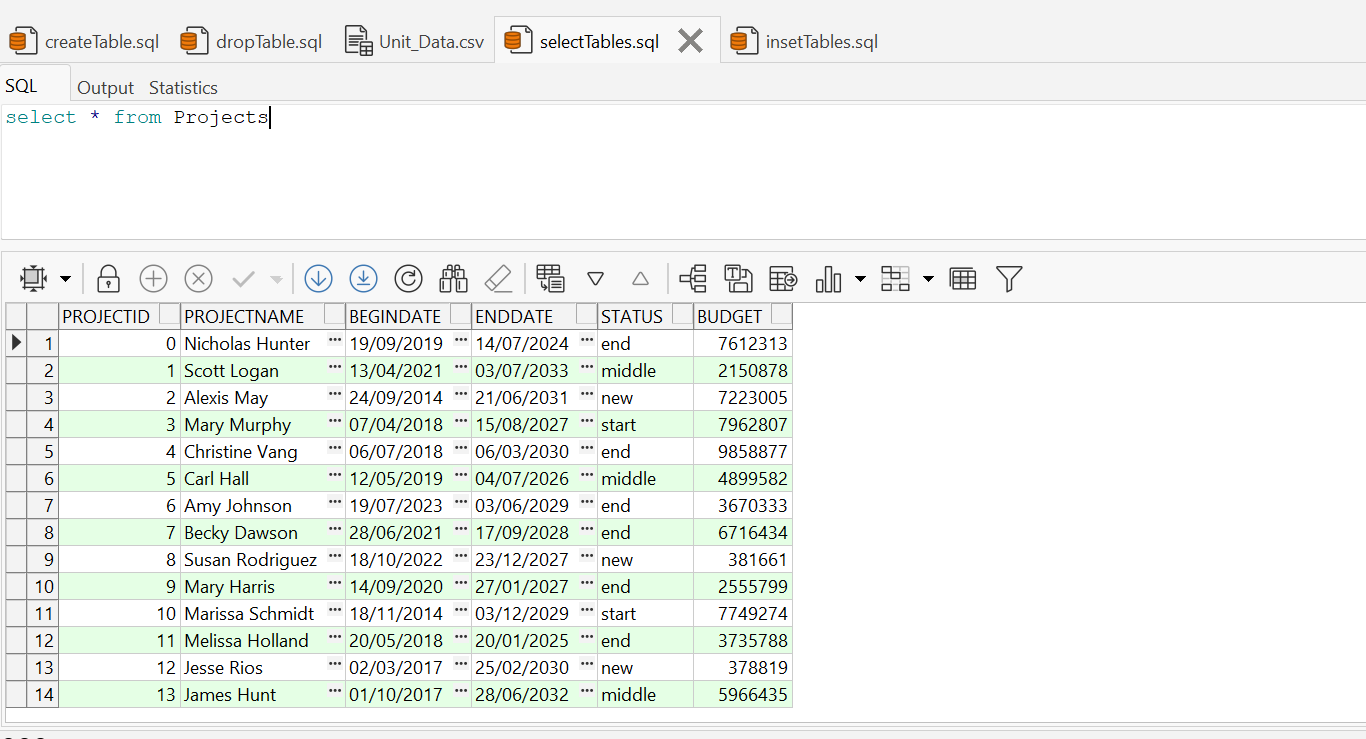
**הטבלה InPriject:**





**דרך שלישית : על ידי יצירת קוד פייתון שמייצר פקודות Insert :**

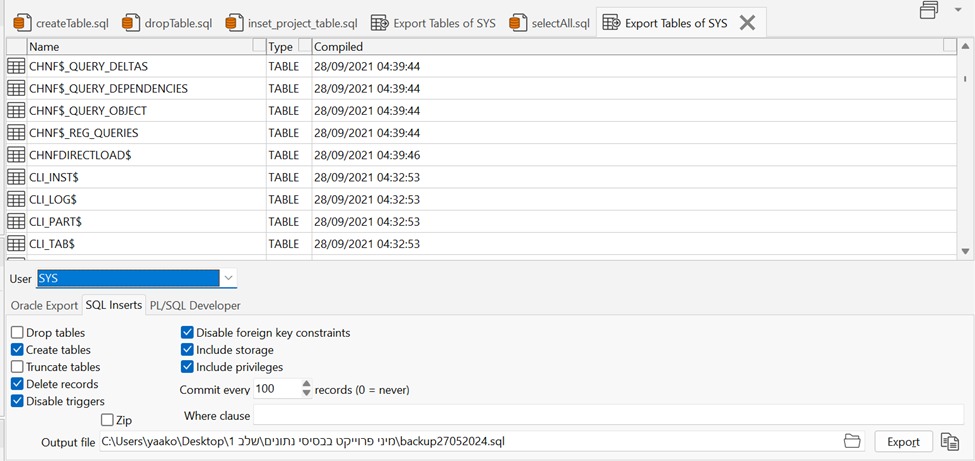
**הטבלה Projects:**

פלט הקוד :

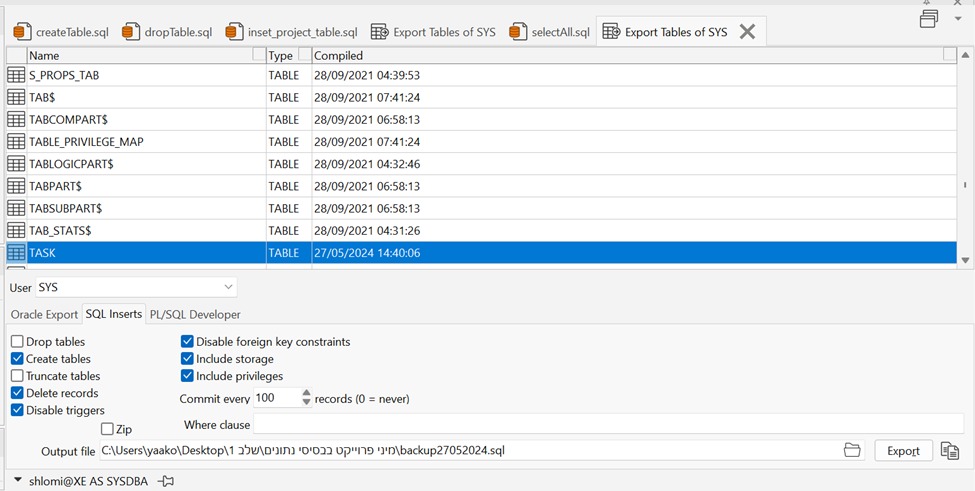
לאחר ההוספה:

* 1. **גיבוי**

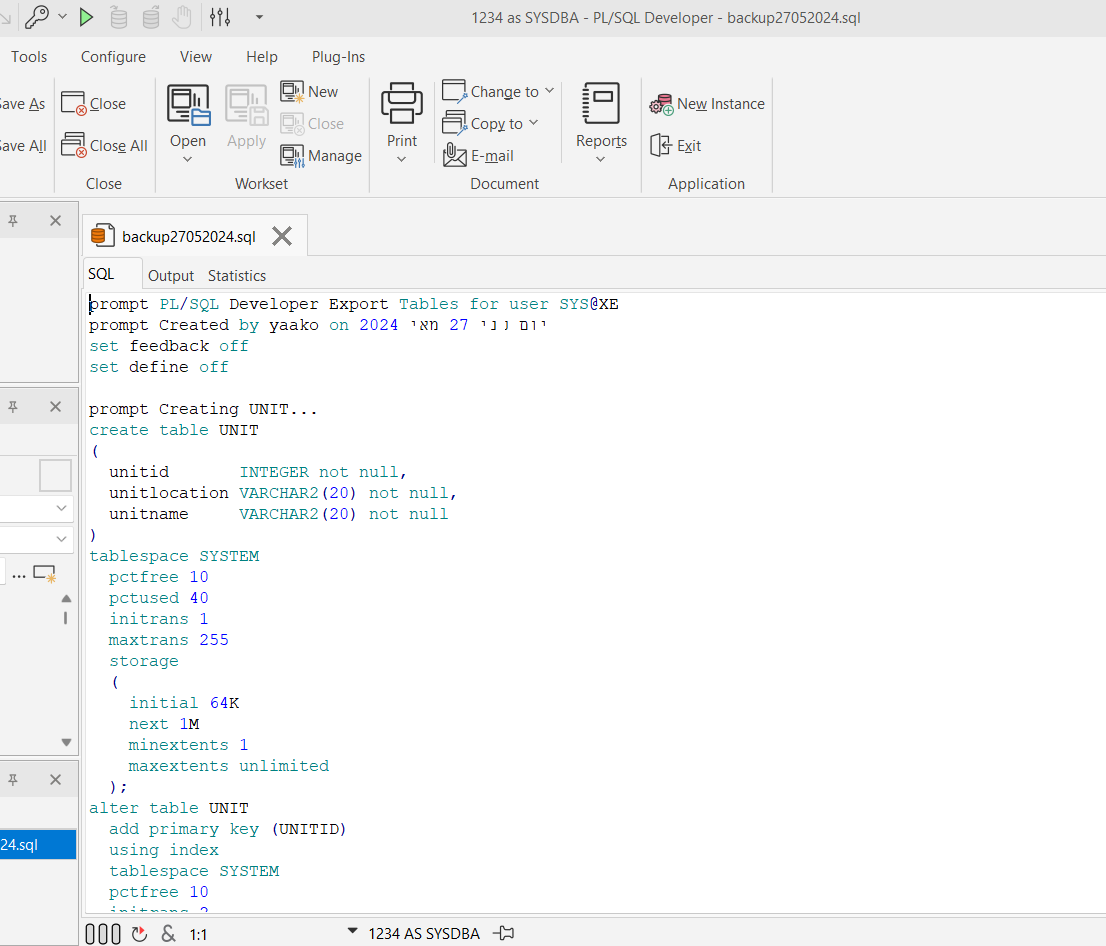
נלחץ על tools->export tables->SQL Insert

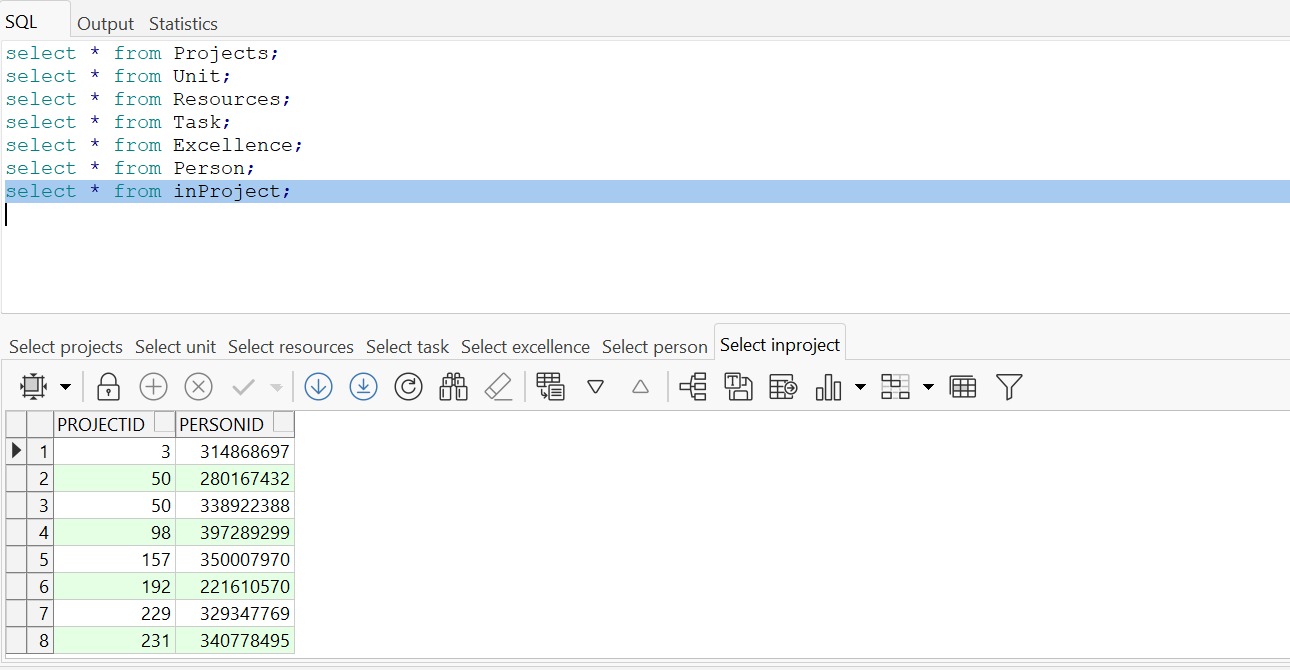


נבחר את הטבלאות שאותן נרצה לגבות (שהן כל הטבלאות שיצרנו) ע"י הקשת קונטרול ואז לחיצה על הטבלאות.



ניתן לקובץ הגיבוי שם שיכלול תאריך כדי שנדע מתי הוא היה, ונלחץ export.

  
  
הרצה של קובץ הגיבוי

לאחר הריצה נעשהSELECT :  


**שלב 2**

* 1. **שאילתות – select ללא פרמטרים**
     1. **קבלת מספר האנשים שהוקצו לכל פרויקט**

**תיאור**

השאילתה מחזירה את מספר האנשים שהוקצו לכל פרויקט.

עבור כל פרויקט היא מחזירה את שם הפרויקט, זמן ההתחלה וזמן הסיום שלו וכן את מספר האנשים שהוקצו לפרויקט.

הפרויקטים מסודרים לפי תאריך סיום הפרויקט בסדר יורד.

**הקוד ב-SQL**

SELECT p.projectName, p.beginDate, p.endDate, COUNT(ip.personID) AS total\_people

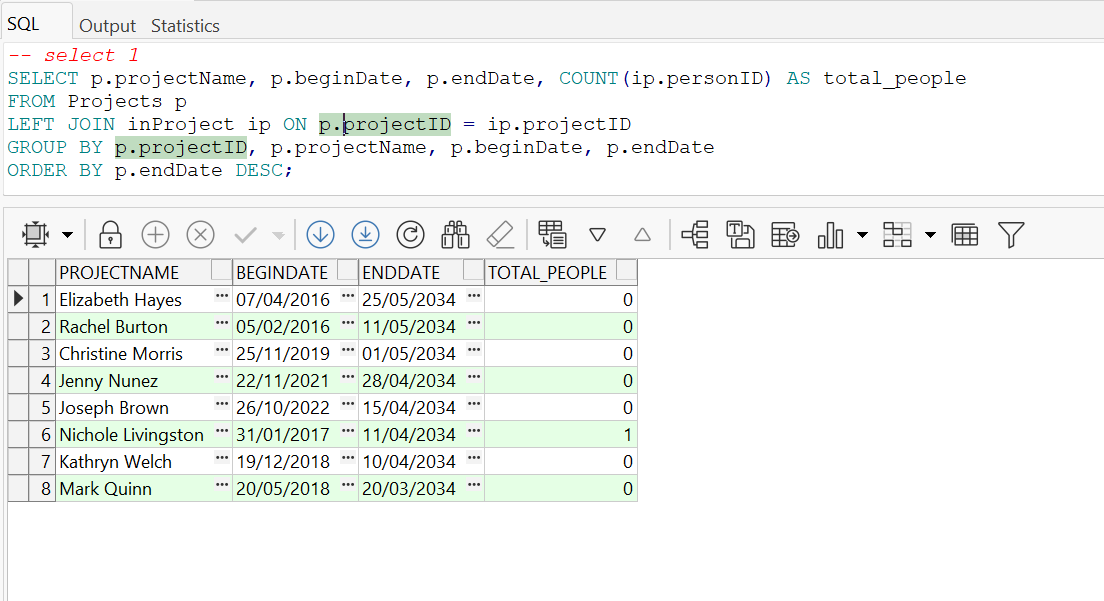
FROM Projects p

LEFT JOIN inProject ip ON p.projectID = ip.projectID

GROUP BY p.projectID, p.projectName, p.beginDate, p.endDate

ORDER BY p.endDate DESC;

**מדגם תוצאות ההרצה**

****

* + 1. **קבלת הפרטים של אנשים שמעורבים ביותר מפרויקט אחד**

**תיאור**

השאילתה מחזירה את השמות, התפקידים ומספר הפרויקטים בהם הם מעורבים, של אנשים שמשתתפים ביותר מפרויקט אחד.

**הקוד ב-SQL**

SELECT p.personName, p.personRole, COUNT(ip.projectID) AS project\_count

FROM Person p

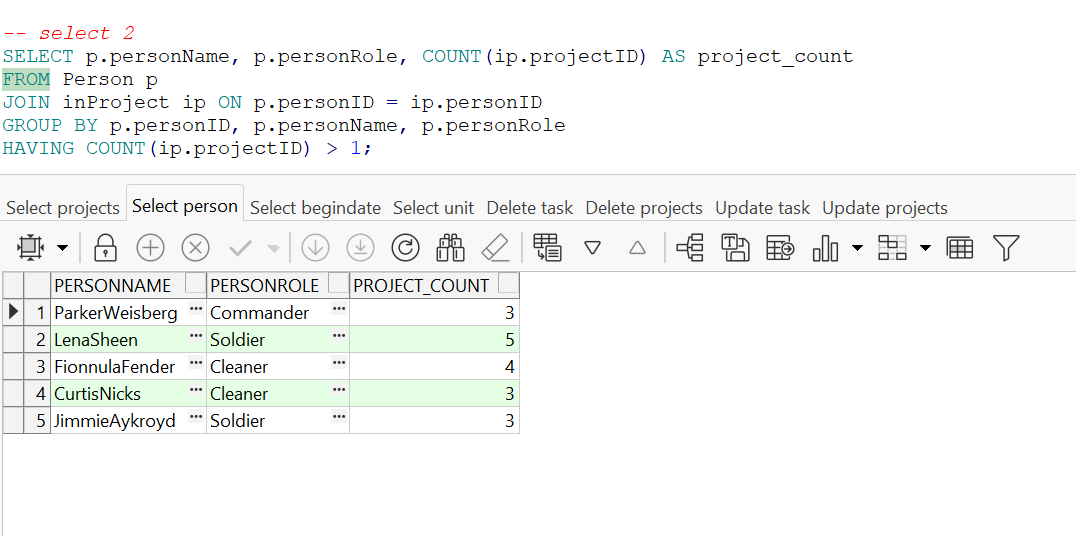
JOIN inProject ip ON p.personID = ip.personID

GROUP BY p.personID, p.personName, p.personRole

HAVING COUNT(ip.projectID) > 1;

**מדגם תוצאות ההרצה**

הערה: הוספנו לטבלה inProject נתונים נוספים על מנת שיהיו אנשים שיעמדו בתנאי השאילתה.

****

* + 1. **קבלת פרטי הפרויקטים כולל מספר המשימות בכל סטטוס**

**תיאור**

השאילתה מחזירה את פרטי כל אחד מהפרויקטים.

עבור כל פרויקט, מחזירה את שם הפרויקט, שנת ההתחלה והסיום, וכמה משימות יש בכל סטטוס ("end", "middle", "start", "new").

**הקוד ב-SQL**

SELECT p.projectName,

EXTRACT(YEAR FROM p.beginDate) AS start\_year,

EXTRACT(YEAR FROM p.endDate) AS end\_year,

SUM(CASE WHEN t.status = 'new' THEN 1 ELSE 0 END) AS new\_tasks,

SUM(CASE WHEN t.status = 'start' THEN 1 ELSE 0 END) AS start\_tasks,

SUM(CASE WHEN t.status = 'middle' THEN 1 ELSE 0 END) AS middle\_tasks,

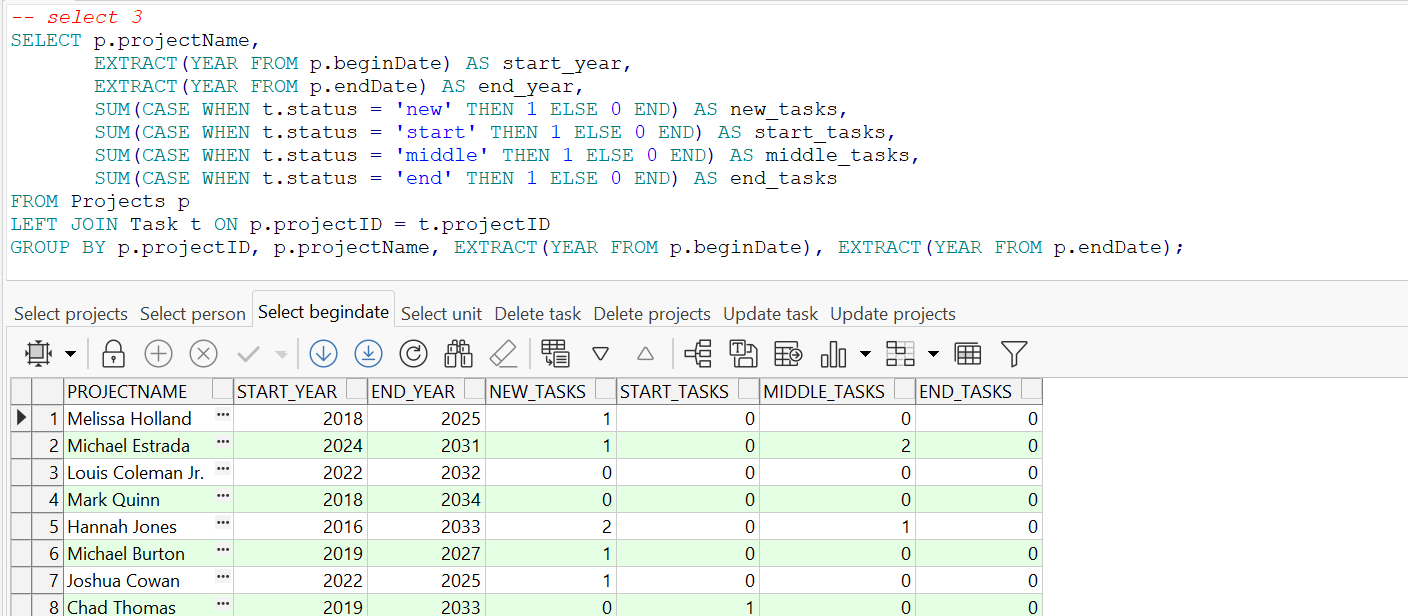
SUM(CASE WHEN t.status = 'end' THEN 1 ELSE 0 END) AS end\_tasks

FROM Projects p

LEFT JOIN Task t ON p.projectID = t.projectID

GROUP BY p.projectID, p.projectName, EXTRACT(YEAR FROM p.beginDate), EXTRACT(YEAR FROM p.endDate);ORDER BY p.endDate DESC;

**מדגם תוצאות ההרצה**

****

* + 1. **קבלת כמות המשאבים והאנשים של היחידות שקיבלו הצטיינות בשנים מסוימות.**

**תיאור**

השאילתה מחזירה את כמות המשאבים וכן כמות האנשים ביחידות שקיבלו הצטיינות לפני שנת 2000.

הסיבה שהשתמשנו ב-left join היא כדי שגם יחידות מצטיינות שאין להן כלל משאבים או אנשים, תתקבלנה.

**הקוד ב-SQL**

SELECT u.unitName, COUNT(DISTINCT r.resourceID) AS total\_resources,

COUNT(DISTINCT p.personID) AS total\_people

FROM Unit u

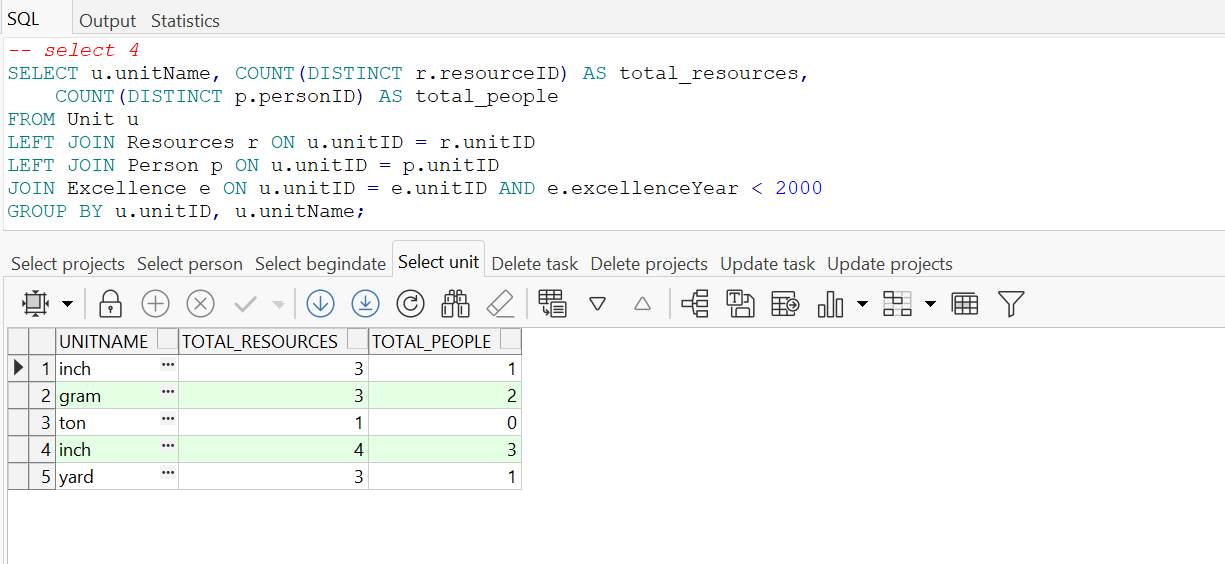
LEFT JOIN Resources r ON u.unitID = r.unitID

LEFT JOIN Person p ON u.unitID = p.unitID

JOIN Excellence e ON u.unitID = e.unitID AND e.excellenceYear < 2000

GROUP BY u.unitID, u.unitName;

**מדגם תוצאות ההרצה**

****

* 1. **שאילתות - delete**
     1. **מחיקת משימות שהסתיימו לפני זמן מסויים.**

**תיאור**

השאילתה מוחקת את המשימות שהסטטוס שלהן הוא "end" ותאריך הסיום שלהן הוא לפני ה-1.1.2023.

**הקוד ב-SQL**

DELETE FROM Task

WHERE status = 'end' AND endDate < TO\_DATE('2023-01-01', 'YYYY-MM-DD');

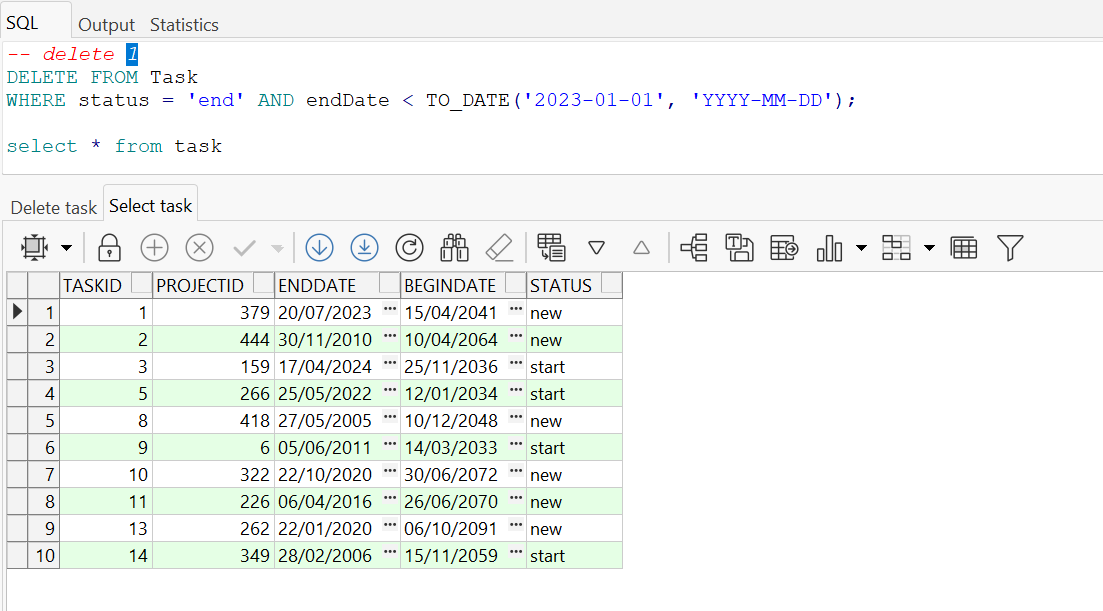
**מדגם תוצאות ההרצה**

**לפני -**

**תמונה שמכילה טקסט, צילום מסך, מספר, גופן

התיאור נוצר באופן אוטומטי**

**אחרי -**

****

כפי שניתן לראות, המשימות שהסתיימו ותאריך הסיום שלהן הוא לפני ה- 1.1.2023, נמחקו.

* + 1. **מחיקת פרויקטים ללא הקצאות.**

**תיאור**

השאילתה מוחקת פרויקטים שאין להם 1. הקצאה של אנשים 2. אין להם אף משימה.

**הקוד ב-SQL**

DELETE FROM Projects p

WHERE p.projectID NOT IN (

SELECT DISTINCT t.projectID

FROM Task t

)

AND p.projectID NOT IN (

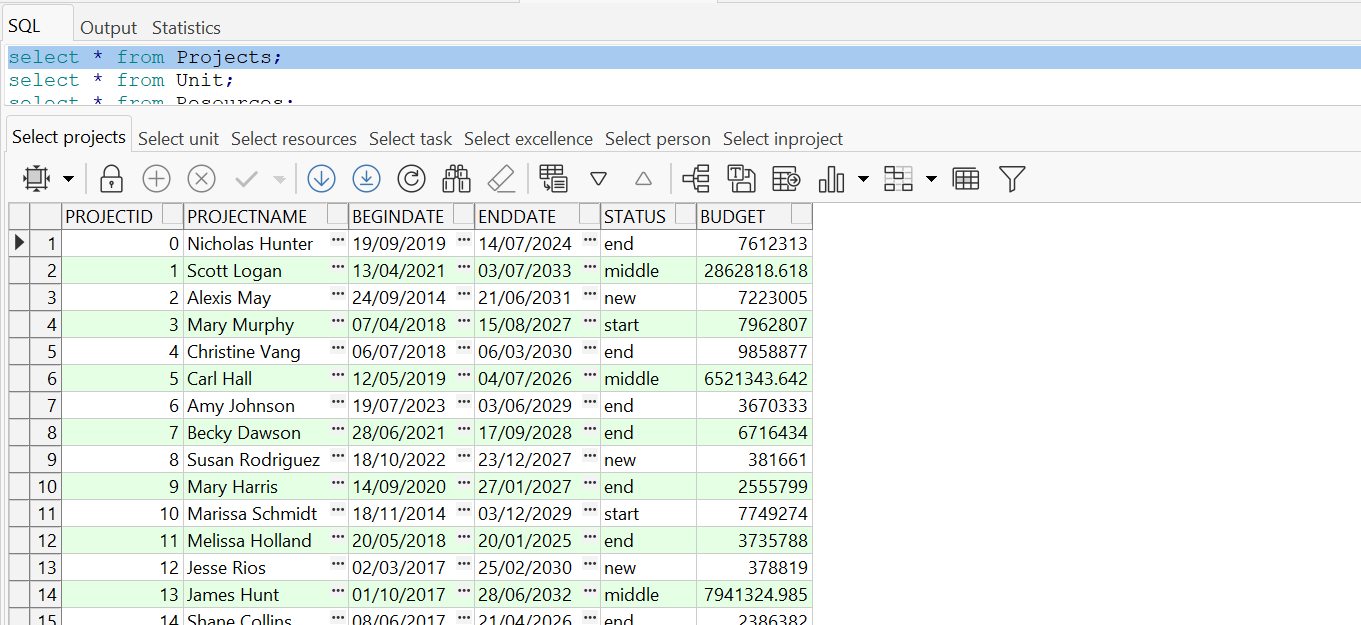
SELECT DISTINCT ip.projectID

FROM inProject ip

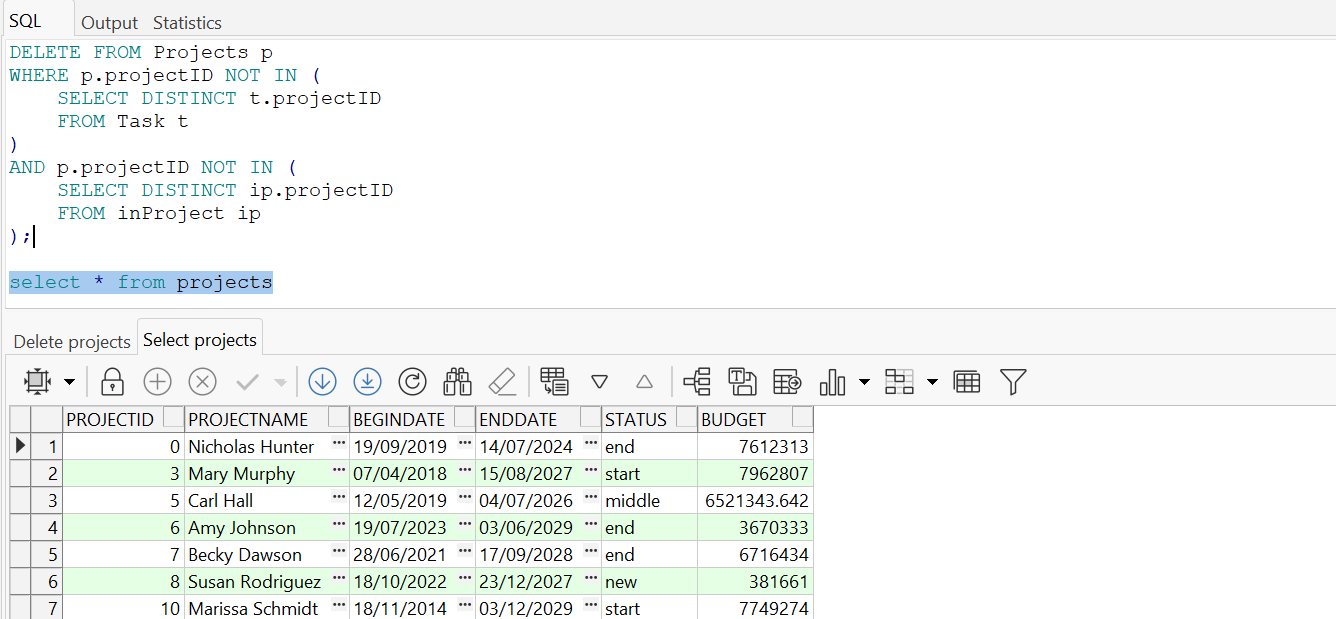
);

**מדגם תוצאות ההרצה**

**לפני -**

****

**אחרי -**

****

כפי שניתן לראות, פרויקטים שאין להם משימות וכן אין אנשים שהוקצו להם, נמחקו.

לדוגמה, הפרויקטים עם ה-projectID 1,2 וכו'.

* 1. **שאילתות - update**
     1. **עדכון הסטטוס של משימות מסוימות.**

**תיאור**

עבור משימה שזמן הסיום שלה הוא בין התאריכים 1.6.2023 ו- 31.8.2023, השאילתה מעדכנת את הסטטוס ל-"end".

**הקוד ב-SQL**

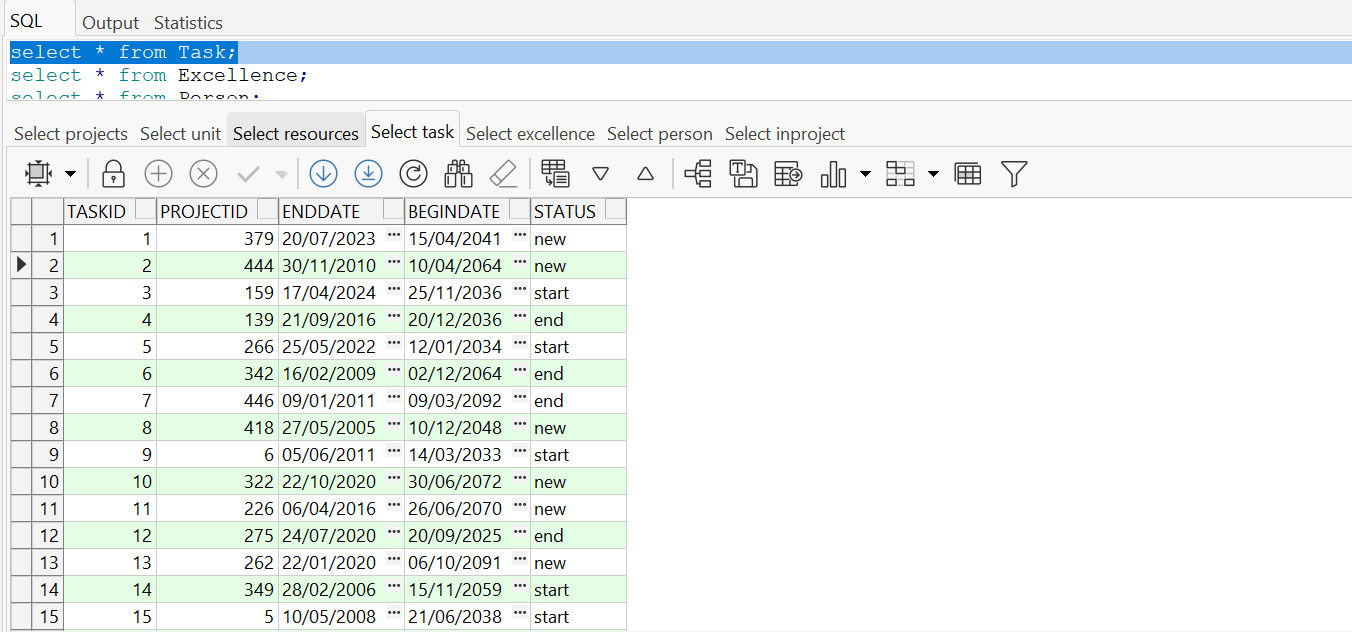
UPDATE Task

SET status = 'end'

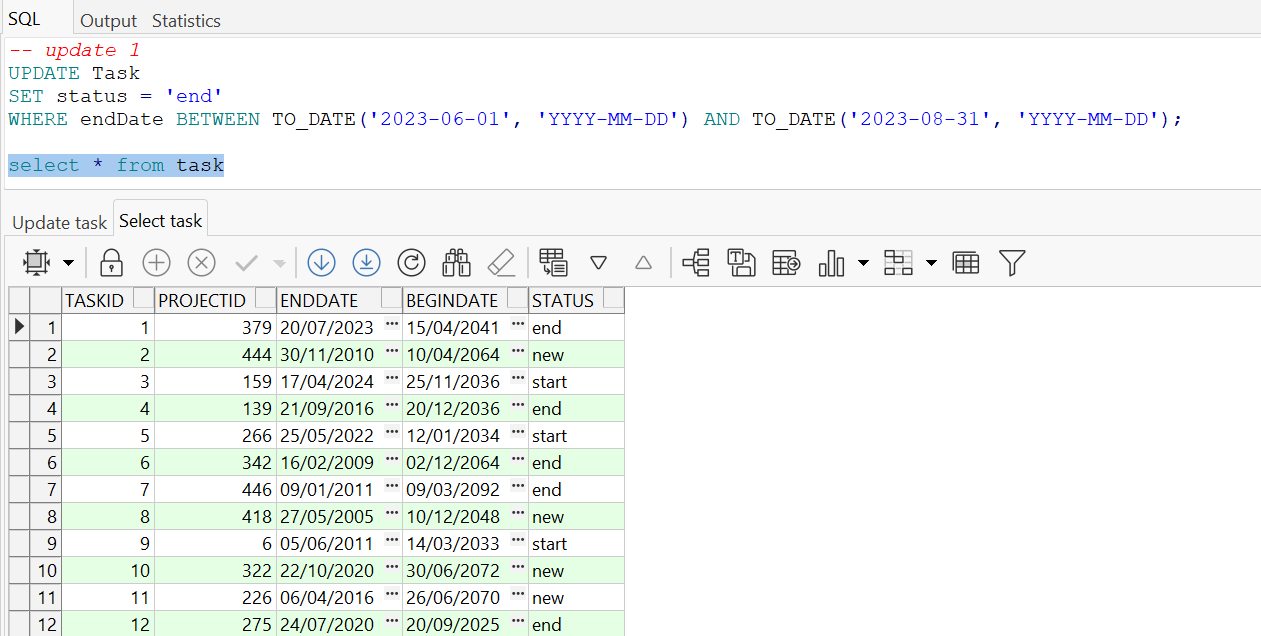
WHERE endDate BETWEEN TO\_DATE('2023-06-01', 'YYYY-MM-DD') AND TO\_DATE('2023-08-31', 'YYYY-MM-DD');

**מדגם תוצאות ההרצה**

**לפני -**

****

**אחרי -**

****

כפי שניתן לראות, המשימות שזמן הסיום שלהן בתאריכים הרלוונטיים, נמחקו.

לדוגמה, המשימה הראשונה בטבלה.

* + 1. **עדכון התקציב של כל הפרויקטים שהתחילו בשנה הנוכחית.**

**תיאור**

עבור פרויקט ששנת ההתחלה שלו היא השנה הנוכחית, השאילתה מגדילה את התקציב שלו ב-10%.

**הקוד ב-SQL**

UPDATE Projects pr

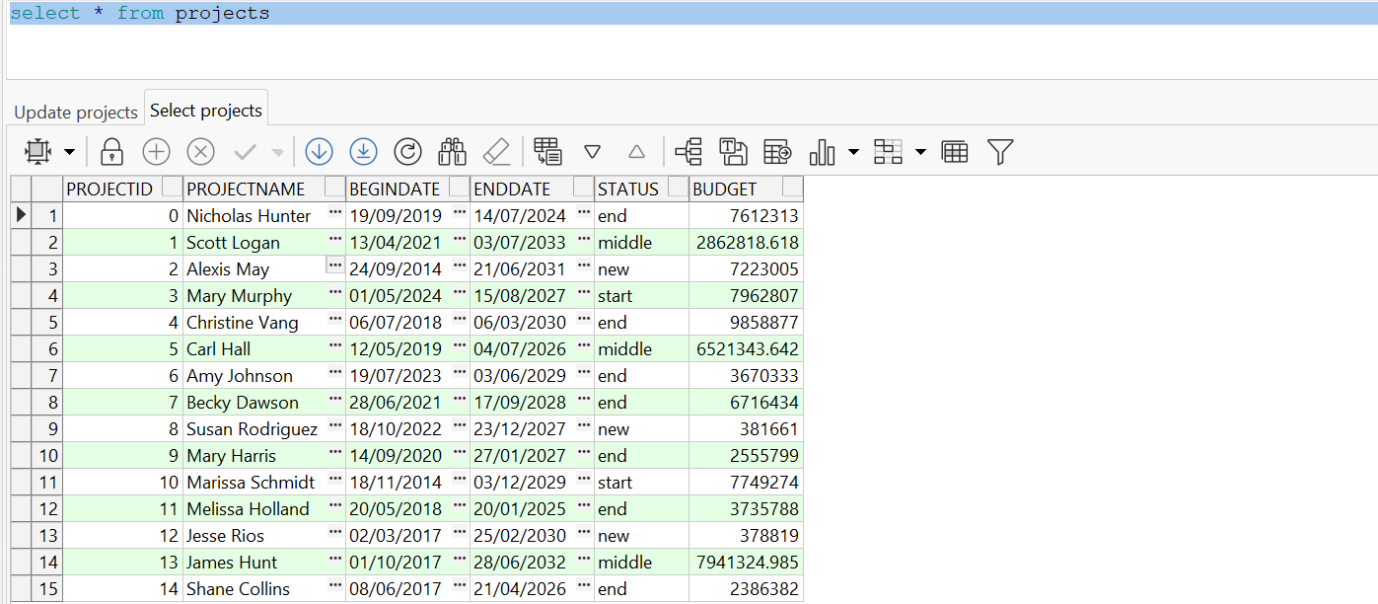
SET pr.budget = pr.budget \* 1.1

WHERE pr.status = 'start' AND EXTRACT(YEAR FROM pr.beginDate) = EXTRACT(YEAR FROM SYSDATE);

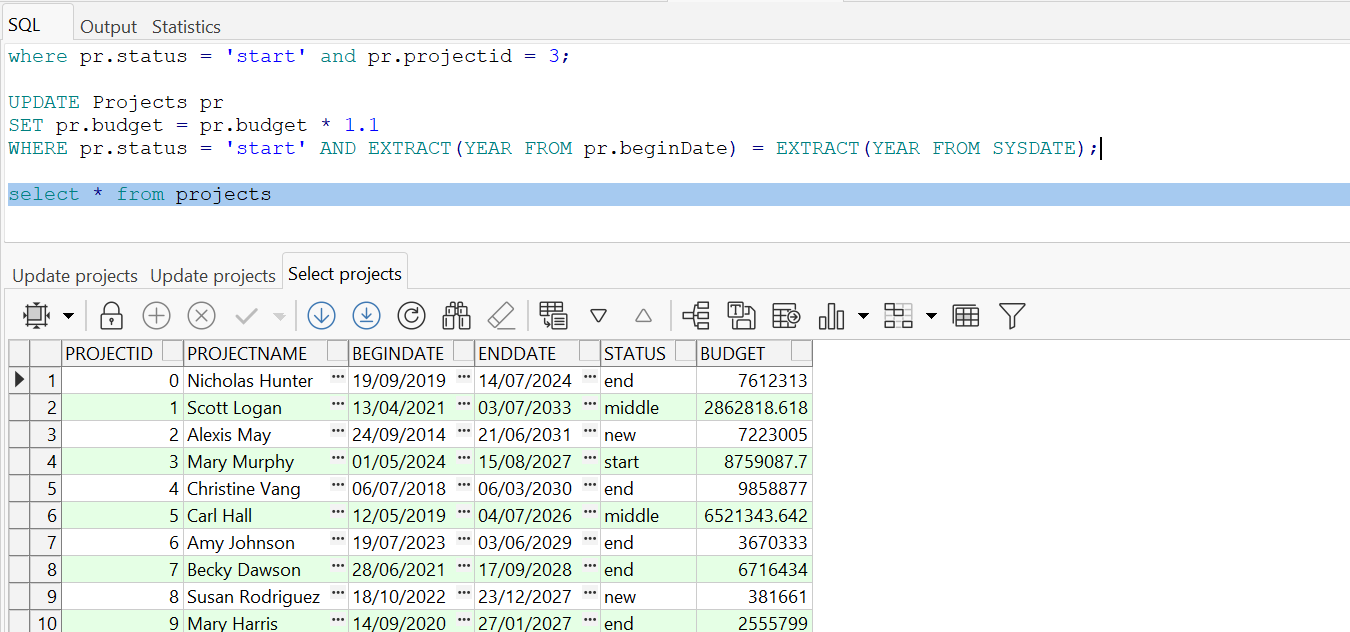
**מדגם תוצאות ההרצה**

**לפני -**

הערה: לא היה בצילום המסך פרויקט שעמד בתנאים, אז עדכנו את התאריך של הפרויקט עם ID 3 לשם ההדגמה

****

**אחרי -**

****

כפי שניתן לראות, התקציב של הפרויקטים בסטטוס "start" שזמן ההתחלה שלהם הוא השנה הנוכחית (2024), הוכפל ב-1.1.

לדוגמה, הפרויקט בעל ID 3.

* 1. **שאילתות – select עם פרמטרים**
     1. **קבלת פרטי האנשים והפרויקטים שהיחידה שאחראית עליהם יש לפחות כמות משאבים מסוימת**

**תיאור**

השאילתה מחזירה את שם האדם, התפקיד שלו ושם הפרויקט, אם ליחידה שאליה משתייך האדם, יש יותר מכמות מסוימת (פרמטר שמוכנס ע"י המשתמש) של פרמטרים.

**הקוד ב-SQL**

SELECT p.personName, p.personRole, pr.projectName

FROM Person p

JOIN inProject ip ON p.personID = ip.personID

JOIN Projects pr ON ip.projectID = pr.projectID

JOIN Task t ON pr.projectID = t.projectID

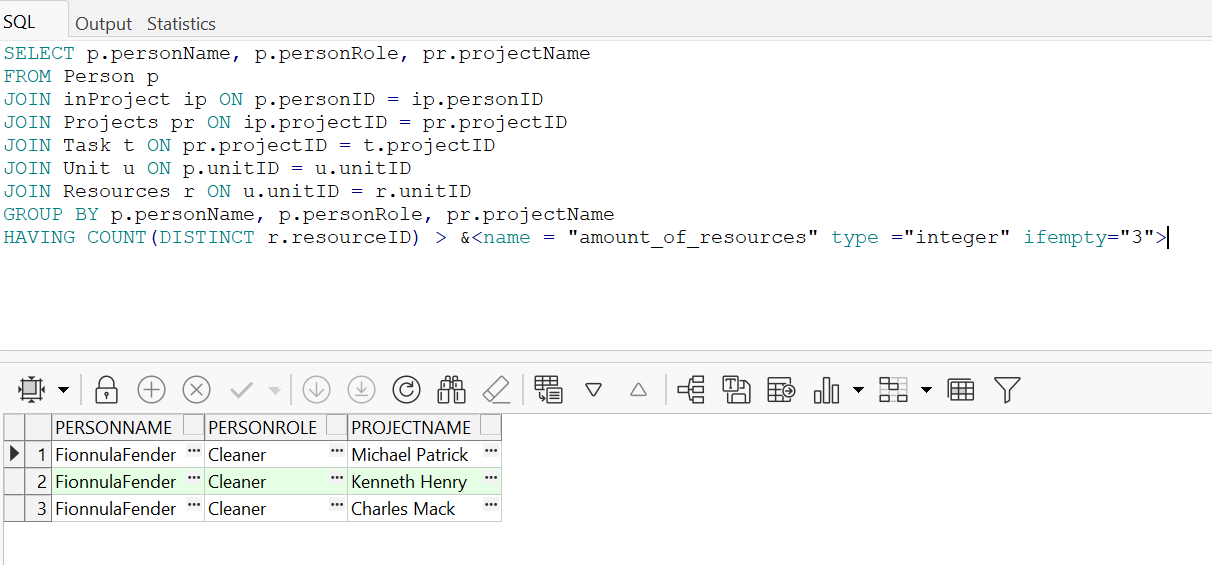
JOIN Unit u ON p.unitID = u.unitID

JOIN Resources r ON u.unitID = r.unitID

GROUP BY p.personName, p.personRole, pr.projectName

HAVING COUNT(DISTINCT r.resourceID) > &<name = "amount\_of\_resources" type ="integer" ifempty="3">

**מדגם תוצאות ההרצה**

****

זאת דוגמת הרצה של השאילתה כשהפרמטר שהכנסנו הוא – 3.

* + 1. **קבלת פרויקטים שהתקציב שלהם נמצא בטווח מסוים**

**תיאור**

השאילתה מחזירה את שם הפרויקט, זמן ההתחלה וזמן הסיום, של כל הפרויקטים שהתקציב שלהם נמצא בטווח שבין min\_budget ו-max\_budget, כאשר המשתנים הללו הם פרמטרים שמתקבלים ע"י המשתמש.

בנוסף, הערך הדיפולטי (אם המשתמש לא מכניס ערך) של min\_budget הוא 2100.

**הקוד ב-SQL**

SELECT projectName, beginDate, endDate

FROM Projects

WHERE budget BETWEEN &<name = "min\_budget"

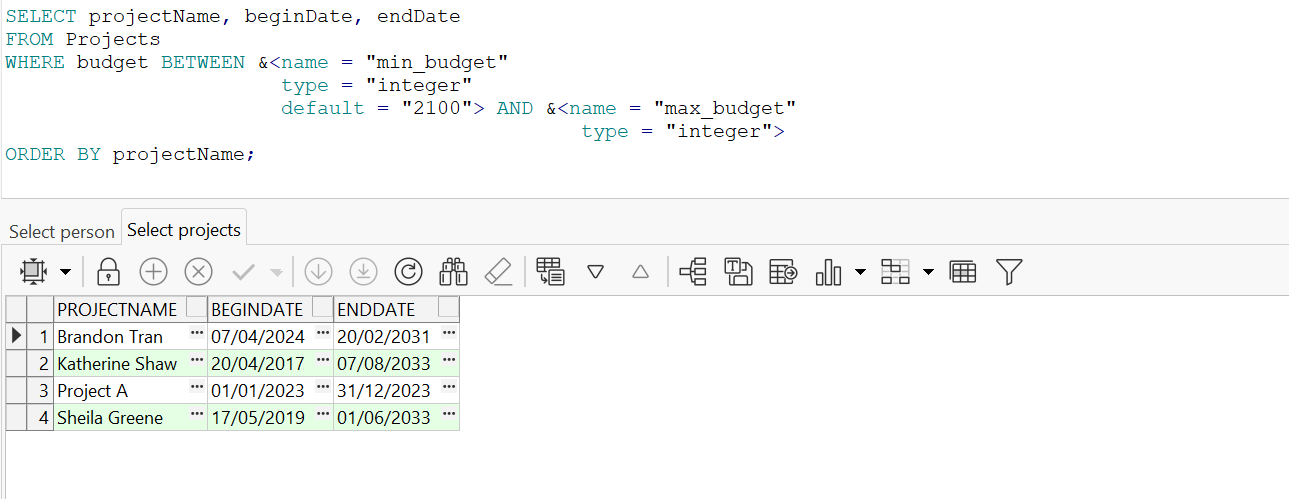
type = "integer"

default = "2100"> AND &<name = "max\_budget"

type = "integer">

ORDER BY projectName;

**מדגם תוצאות ההרצה**

****

זאת דוגמת הרצה של השאילתה כשהפרמטר max\_budget שהכנסנו הוא – 10000 והפרמטר min\_budget נשאר דיפולטי.

* + 1. **קבלת כל הפרויקטים שהסטטוס שלהם הוא ברשימה שנבחרה והתקציב שלהם נמצא בטווח מסוים**

**תיאור**

השאילתה מחזירה את כל הפרויקטים שהתקציבים שלהם נמצאים בטווח שבין min\_budget ו-max\_budget ושהסטטוס שלהם הוא אחד מהסטטוסים שנבחרו ע"י המשתמש מתוך ה-list, ממוינים לפי שם הפרויקט בסדר אלפביתי.

השאילתה מחזירה את שם הפרויקט, זמן ההתחלה וזמן הסיום שלו.

**הקוד ב-SQL**

SELECT projectName, beginDate, endDate

FROM Projects p

WHERE p.budget BETWEEN &<name = "min\_budget"

type = "integer"

default = "2100"> AND &<name = "max\_budget"

type = "integer">

AND p.status IN (&<name = "status"

type = "string"

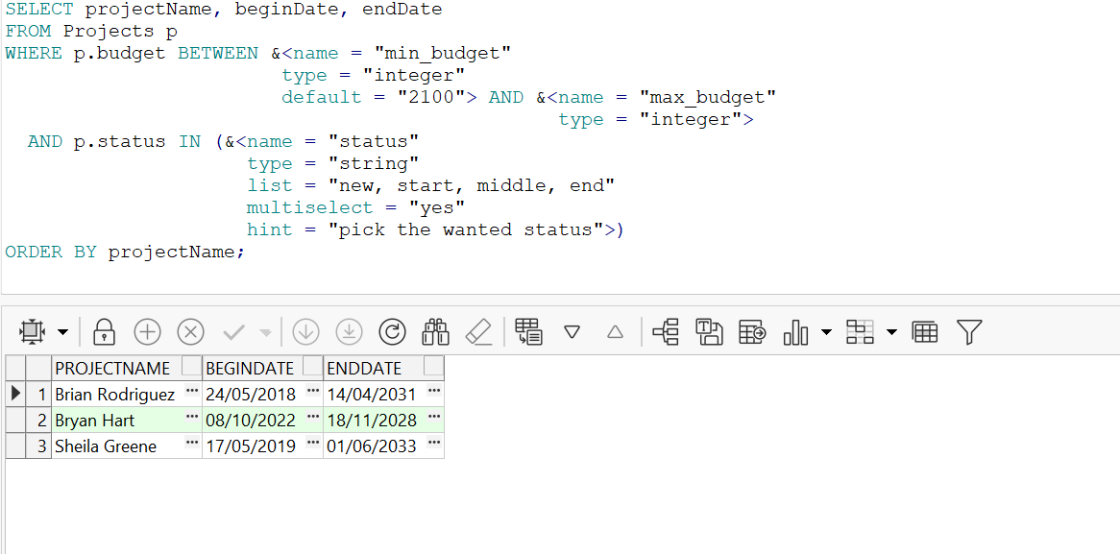
list = "new, start, middle, end"

multiselect = "yes"

hint = "pick the wanted status">)

ORDER BY projectName;

**מדגם תוצאות ההרצה**

****

זאת דוגמת הרצה של השאילתה כשהפרמטר max\_budget שהכנסנו הוא – 100000, הפרמטר min\_budget הוא 1000 ובחרנו מהרשימה של הסטטוס רק את end.

* + 1. **קבלת כל הפרויקטים בסטטוס שהמשתמש בוחר**

**תיאור**

השאילתה מחזירה את הפרויקטים שעומדים בסטטוס שהמשתמש הכניס וממיינת אותם לפי תאריך הסיום של הפרויקט בסדר יורד.

השאילתה מחזירה את שם הפרויקט, זמן ההתחלה, זמן הסיום שלו וכמות האנשים שעובדים עליו.

**הקוד ב-SQL**

SELECT p.projectName, p.beginDate, p.endDate, COUNT(ip.personID) AS total\_people

FROM Projects p

LEFT JOIN inProject ip ON p.projectID = ip.projectID

Where p.status = &<name ="status"

type="string"

list ="start, middle, end"

multiselect ="yes"

hint ="pick the wanted status">

GROUP BY

p.projectID,

p.projectName,

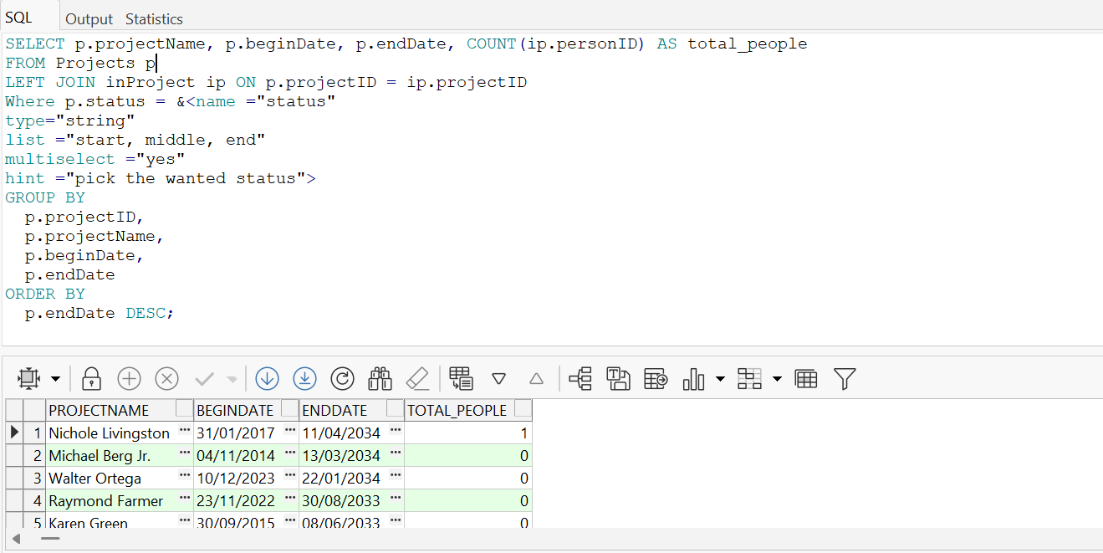
p.beginDate,

p.endDate

ORDER BY

p.endDate DESC;

**מדגם תוצאות ההרצה**

****

זאת דוגמת הרצה של השאילתה כשהפרמטר status הוא end.

* 1. **אילוצים**
     1. **אילוץ מסוג NOT NULL בטבלה person**

**תיאור**

העמודה personRank בטבלה Person משתנה, כך שתכלול אילוץ NOT NULL.

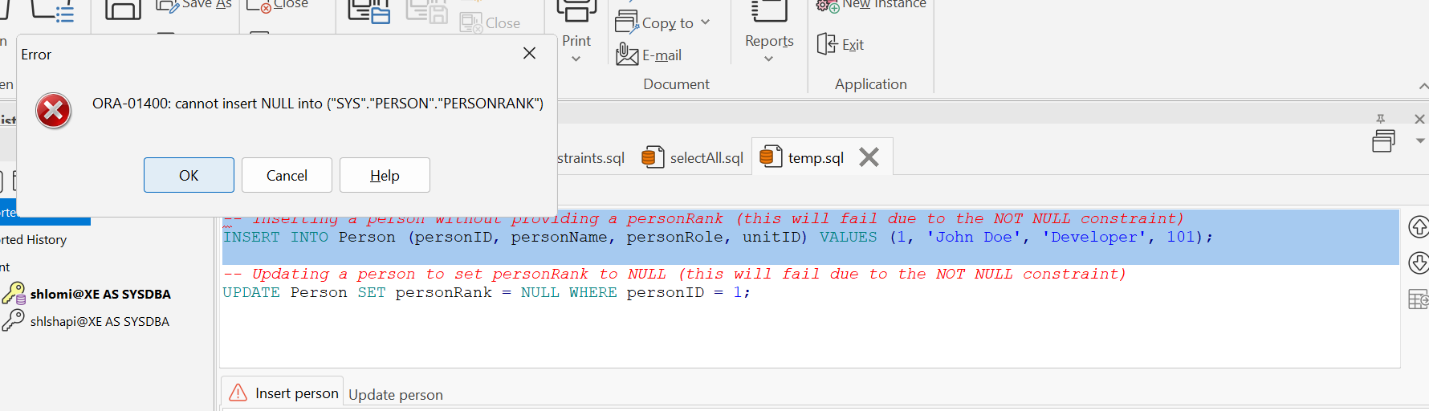
האילוץ מחייב שלכל רשומה בטבלה person יהיה ערך לא ריק עבור העמודה personRank.

אם ננסה להכניס או לעדכן רשומה, בלי לספק ערך עבור personRank, תהיה לשגיאה.

**הקוד ב-SQL**

ALTER TABLE Person MODIFY personRank VARCHAR2(20) NOT NULL;

**ניסיון הרצה שסותרת את האילוץ**

****

* + 1. **אילוץ מסוג CHECK בטבלה resources**

**תיאור**

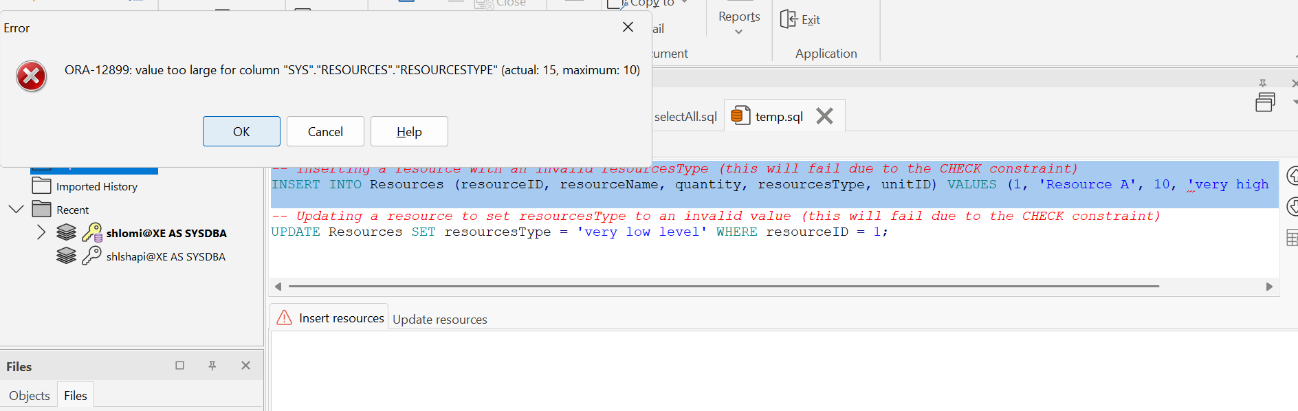
האילוץ CHECK מתווסף לעמודה resourcesType בטבלה resources, כדי להבטיח שהיא יכולה להכיל רק ערכים ספציפיים: 'low level', 'medium lev' או 'high level'.

האילוץ הזה בעצם אוכף את שלמות הנתונים על ידי הגבלת הערכים המותרים עבור resourcesType.

**הקוד ב-SQL**

ALTER TABLE Resources ADD CONSTRAINT resourcesType CHECK (resourcesType IN ('low level', 'medium lev', 'high level'));

**ניסיון הרצה שסותרת את האילוץ**

****

* + 1. **אילוץ מסוג DEFAULT בטבלה projects**

**תיאור**

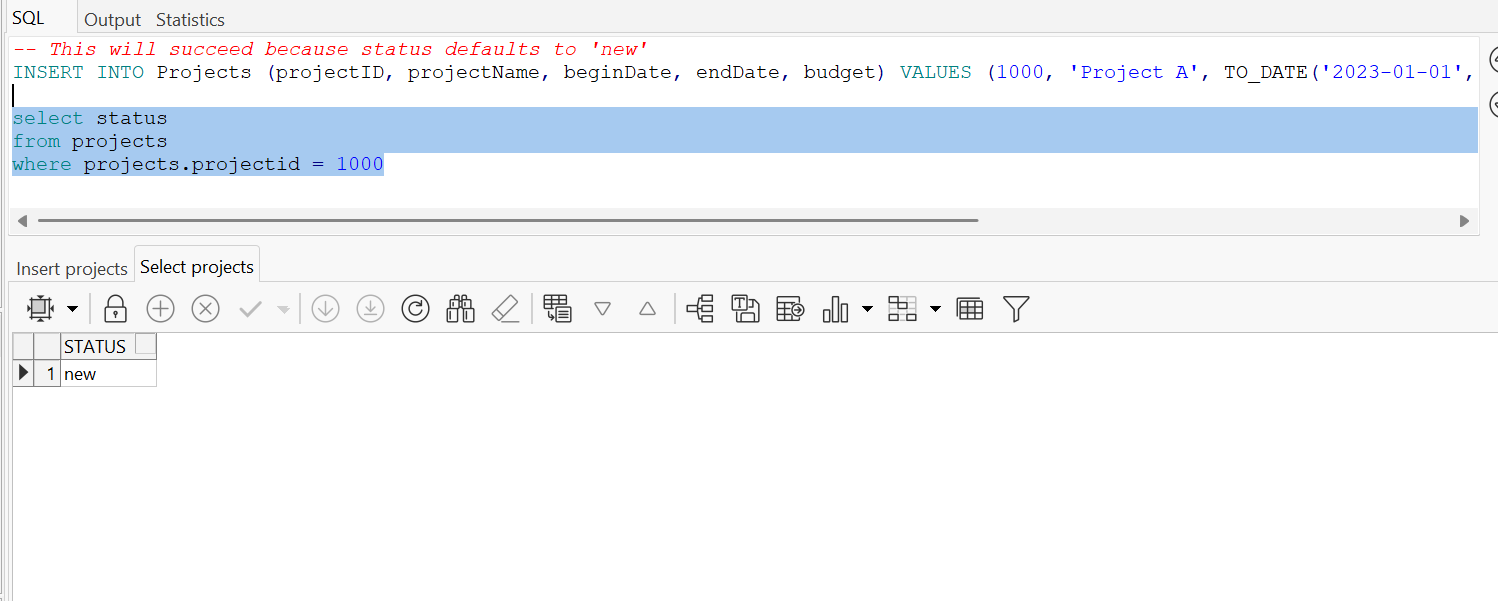
עמודת המצב בטבלת הפרויקטים תשתנה, כך שתכלול אילוץ DEFAULT עם הערך 'new' ואילוץ NOT NULL.

האילוץ הזה מבטיח שאם לא יסופק ערך עבור העמודה status במהלך הוספת פרויקט, ברירת המחדל היא 'חדש'. בנוסף, הוא אוכף שעמודת הסטטוס לא יכולה להיות ריק.

**הקוד ב-SQL**

ALTER TABLE Projects MODIFY status VARCHAR2(10) DEFAULT 'new' NOT NULL;

**הדגמת האילוץ (אי אפשר להראות כאן הפרה)**

כתבנו כאן שאילתה של הכנסת ערך לטבלה projects כאשר לא הכנסנו ערך לשדה status.

כפי שרואים, האילוץ מתקיים והערך של הפרויקט שהכנסנו בשדה status הוא 'new'.

**שלב 3**

* 1. **פרוצדורות**
     1. **הפרוצדורה - CreateNewProjectAndAddPersons**

**מטרה**

יצירת פרויקט חדש וצירוף אנשים לפרויקט.

**תיאור**

קלט:

* + - קורסור של אנשים עבור פרויקט חדש.
    - שם הפרויקט החדש.

פלט:

* + - יצירת פרויקט חדש.
    - צירוף האנשים הנ"ל לפרויקט (כלומר, לטבלה InProject).
    - הדפסת פרטי האנשים והפרויקט החדש.

**הקוד**

CREATE OR REPLACE PROCEDURE CreateNewProjectAndAddPersons(

cur IN SYS\_REFCURSOR,

v\_projectName IN VARCHAR2

) AS

v\_projectID NUMBER;

v\_personID Person.personID%TYPE;

v\_personName Person.personName%TYPE;

v\_personRole Person.personRole%TYPE;

v\_personRank Person.personRank%TYPE;

v\_unitID Person.unitID%TYPE;

BEGIN

*-- Calculate the new project ID*

SELECT NVL(MAX(projectID), 0) + 1 INTO v\_projectID FROM Projects;

*-- Insert new project record*

INSERT INTO Projects (projectID, projectName, beginDate, endDate, status, budget)

VALUES (v\_projectID, v\_projectName, SYSDATE, ADD\_MONTHS(SYSDATE, 18), 'start', 0);

*-- Insert persons into the new project*

LOOP

FETCH cur INTO v\_personID, v\_personName, v\_personRole, v\_personRank, v\_unitID;

EXIT WHEN cur%NOTFOUND;

INSERT INTO inProject (projectID, personID) VALUES (v\_projectID, v\_personID);

DBMS\_OUTPUT.PUT\_LINE('Person ID: ' || v\_personID || ', Name: ' || v\_personName ||

', Role: ' || v\_personRole || ', Rank: ' || v\_personRank);

END LOOP;

*-- Print confirmation*

DBMS\_OUTPUT.PUT\_LINE('New project created with ID ' || v\_projectID || ' and name ' || v\_projectName);

EXCEPTION

WHEN OTHERS THEN

DBMS\_OUTPUT.PUT\_LINE('Error occurred: ' || SQLERRM);

RAISE; *-- Re-raise the exception for further handling*

END;

**תוצאות ההרצה**

מכיוון שהפרוצדורה מקבלת קורסור, זה קצת קשה להריץ אותה בפני עצמה ולהראות את התוצאות.

נדגים את תוצאות ההרצה כאשר נראה את התוכנית שקוראת לפרוצדורה.

* + 1. **הפרוצדורה - AllocateResourcesForUnit**

**מטרה**

הפרוצדורה מקצה משאבים ליחידה ספציפית לפי מספר הפרויקטים שהאנשים ביחידה עובדים עליהם וההצטיינות של היחידה (האם היא מצטיינת או לא).

**תיאור**

קלט:

* ה-ID של היחידה שעבורה צריך להקצות משאבים (p\_unitID).
* סוג המשאב שיש להקצות (p\_resourcesType).

פלט:

* הפרוצדורה מעדכנת את טבלת המשאבים ומדפיסה הודעות על תהליך ההקצאה (לפי נוסחה של כמות הפרויקטים כפול 5 ואם יש הצטיינות, אז + 10 אחוזים).

**הקוד**

CREATE OR REPLACE PROCEDURE AllocateResourcesForUnit(

p\_unitID IN Unit.unitID%TYPE,

p\_resourcesType IN Resources.resourcesType%TYPE

) IS

TYPE unit\_info\_rec IS RECORD (

unitID Unit.unitID%TYPE,

projectCount NUMBER,

isExcellent CHAR(1)

);

v\_unit\_info unit\_info\_rec;

v\_allocation\_amount NUMBER;

v\_current\_quantity NUMBER;

v\_update\_amount NUMBER;

CURSOR c\_resources IS

SELECT resourceID, quantity

FROM Resources

WHERE unitID = p\_unitID AND resourcesType = p\_resourcesType

FOR UPDATE;

insufficient\_resources EXCEPTION;

BEGIN

*-- Get unit information*

SELECT u.unitID, COUNT(DISTINCT ip.projectID), NVL(MAX(e.have\_money), 'F')

INTO v\_unit\_info

FROM Unit u

LEFT JOIN Person p ON u.unitID = p.unitID

LEFT JOIN inProject ip ON p.personID = ip.personID

LEFT JOIN Excellence e ON u.unitID = e.unitID AND e.excellenceYear = EXTRACT(YEAR FROM SYSDATE)

WHERE u.unitID = p\_unitID

GROUP BY u.unitID;

*-- Calculate allocation amount*

v\_allocation\_amount := FLOOR(v\_unit\_info.projectCount \* 5);

*-- Add 10% if the unit is excellent*

IF v\_unit\_info.isExcellent = 'T' THEN

v\_allocation\_amount := v\_allocation\_amount \* 1.1;

END IF;

DBMS\_OUTPUT.PUT\_LINE('Allocating ' || v\_allocation\_amount || ' resources to Unit ' || p\_unitID);

*-- Allocate resources*

FOR r\_resource IN c\_resources LOOP

v\_current\_quantity := r\_resource.quantity;

UPDATE Resources

SET quantity = quantity + v\_update\_amount

WHERE CURRENT OF c\_resources;

v\_allocation\_amount := v\_allocation\_amount - v\_update\_amount;

EXIT WHEN v\_allocation\_amount <= 0;

END LOOP;

IF v\_allocation\_amount > 0 THEN

RAISE insufficient\_resources;

END IF;

COMMIT;

DBMS\_OUTPUT.PUT\_LINE('Resource allocation completed for Unit ' || p\_unitID);

EXCEPTION

WHEN insufficient\_resources THEN

DBMS\_OUTPUT.PUT\_LINE('Error: Insufficient resources for Unit ' || p\_unitID);

ROLLBACK;

WHEN OTHERS THEN

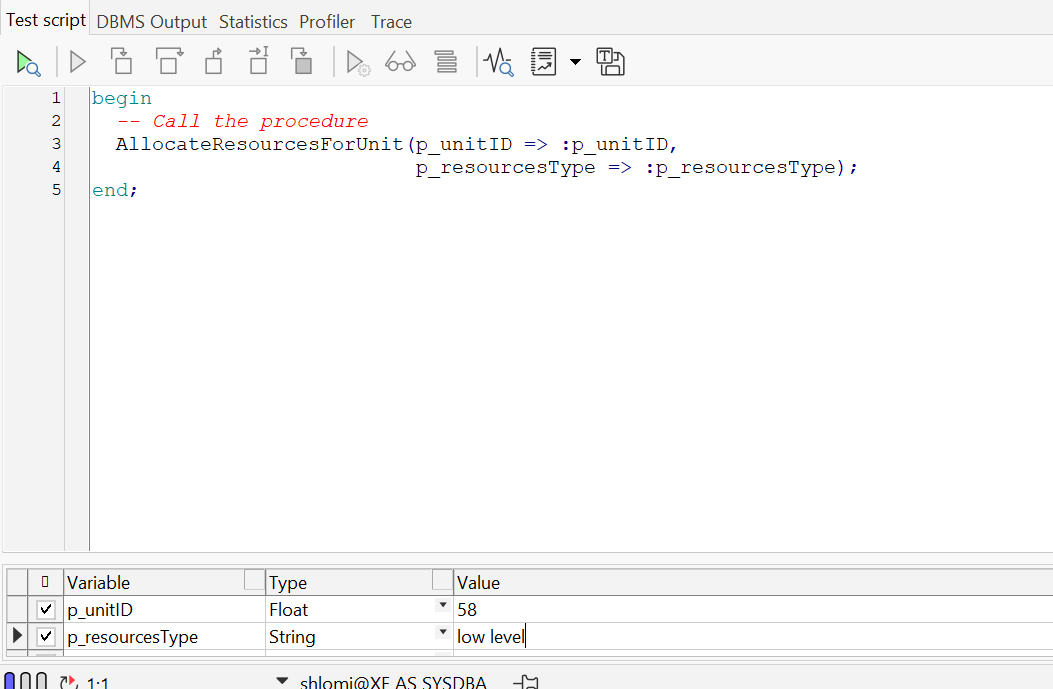
DBMS\_OUTPUT.PUT\_LINE('Error in AllocateResourcesForUnit: ' || SQLERRM);

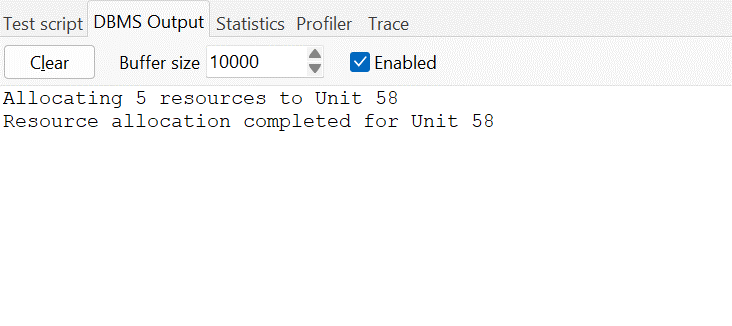
ROLLBACK;

END;

**תוצאות ההרצה**

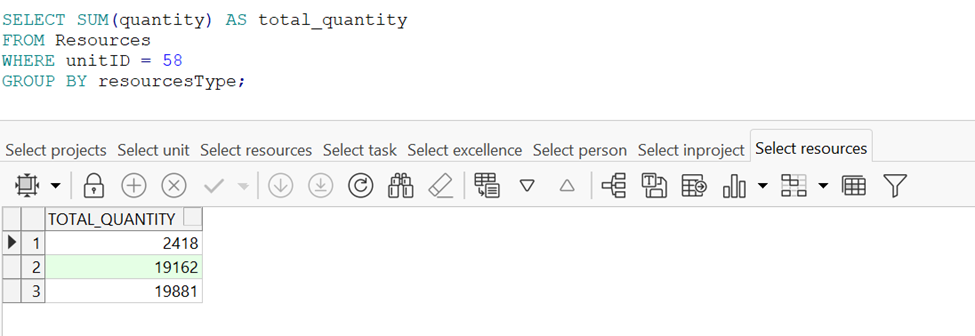
נריץ קובץ טסט עבור הפרוצדורה הזאת עבור יחידה מספר 58 (בדקנו מראש שיש לה פרויקטים) -



 נראה את ההדפסה של הפונקציה -

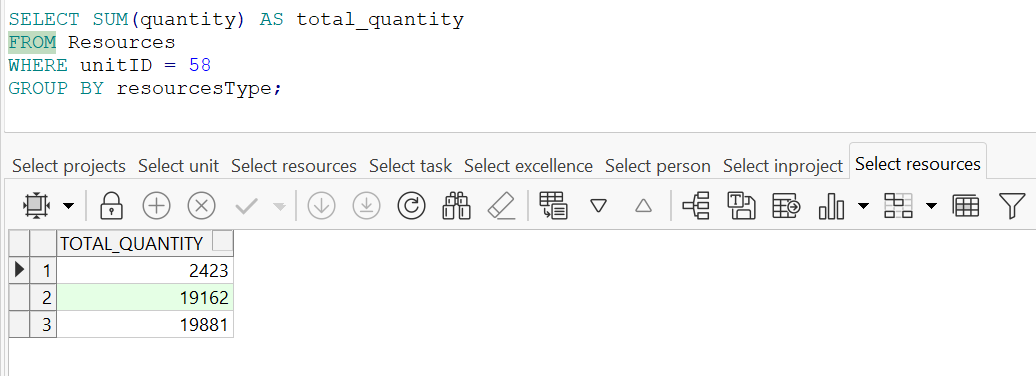
נכתוב פקודת select ונראה שכמות המשאבים של היחידה הנ"ל אכן השתנתה -

זאת תוצאת הרצת השאילתה לפני ביצוע הפרוצדורה -





וזה לאחר ביצוע הפרוצדורה -





* 1. **פונקציות**
     1. **הפונקציה - GetTopNPersons**

**מטרה**

אנחנו רוצים להרים פרויקט חדש ולכן אנחנו מחפשים אנשים מצטיינים (שהגיעו מיחידות שקיבלו הצטיינות), שאינם עובדים כרגע על פרויקטים אחרים.

**תיאור**

קלט:

* + - מספר N שהוא כמות האנשים המבוקשים עבור הפרויקט החדש.

פלט:

הפונקציה מבצעת שאילתה שבודקת האם יש N אנשים שעומדים בקריטריונים הנ"ל.

* + - אם יש לפחות 3/4 מכמות האנשים שצריך עבור הפרויקט, הפונקציה מחזירה קורסור לשאילתה הזאת.

אם יש יותר מ-N, בוחרים את N האנשים שהיחידות שלהם קיבלו את מספר ההצטיינויות הגבוה ביותר.

* אם יש פחות מ-3/4 מהאנשים שצריך עבור הפרויקט, הפונקציה מחזירה NULL.

**הקוד**

CREATE OR REPLACE FUNCTION GetTopNPersons(N IN NUMBER)

RETURN SYS\_REFCURSOR IS

cur SYS\_REFCURSOR;

total\_count INTEGER;

BEGIN

SELECT COUNT(\*)

INTO total\_count

FROM

Person p

WHERE

(NOT EXISTS (

SELECT 1

FROM inProject ip

JOIN Projects pr ON ip.projectID = pr.projectID

WHERE

ip.personID = p.personID

AND pr.status != 'end'

)

OR NOT EXISTS (

SELECT 1

FROM inProject ip

WHERE

ip.personID = p.personID

))

AND p.unitID IN (

SELECT unitID

FROM (

SELECT

unitID,

COUNT(\*) AS excellenceCount

FROM

Excellence

GROUP BY

unitID

ORDER BY

excellenceCount DESC,

unitID ASC

)

);

IF total\_count < CEIL(N \* 0.75) THEN

DBMS\_OUTPUT.PUT\_LINE('Not enough persons to create a new project - there are ' || total\_count || ' from ' || N);

RETURN NULL;

ELSE

OPEN cur FOR

SELECT

p.personID,

p.personName,

p.personRole,

p.personRank,

p.unitID

FROM

Person p

WHERE

(NOT EXISTS (

SELECT 1

FROM inProject ip

JOIN Projects pr ON ip.projectID = pr.projectID

WHERE

ip.personID = p.personID

AND pr.status != 'end'

)

OR NOT EXISTS (

SELECT 1

FROM inProject ip

WHERE

ip.personID = p.personID

))

AND p.unitID IN (

SELECT unitID

FROM (

SELECT

unitID,

COUNT(\*) AS excellenceCount

FROM

Excellence

GROUP BY

unitID

ORDER BY

excellenceCount DESC,

unitID ASC

)

)

ORDER BY

(SELECT COUNT(\*)

FROM Excellence e

WHERE e.unitID = p.unitID) DESC,

p.unitID ASC

FETCH FIRST N ROWS ONLY;

IF total\_count >= N THEN

DBMS\_OUTPUT.PUT\_LINE('Found ' || total\_count || ' persons, but the top ' || N || ' requested.');

ELSE

DBMS\_OUTPUT.PUT\_LINE('Found ' || total\_count || ' persons out of ' || N || ' were selected.');

END IF;

RETURN cur;

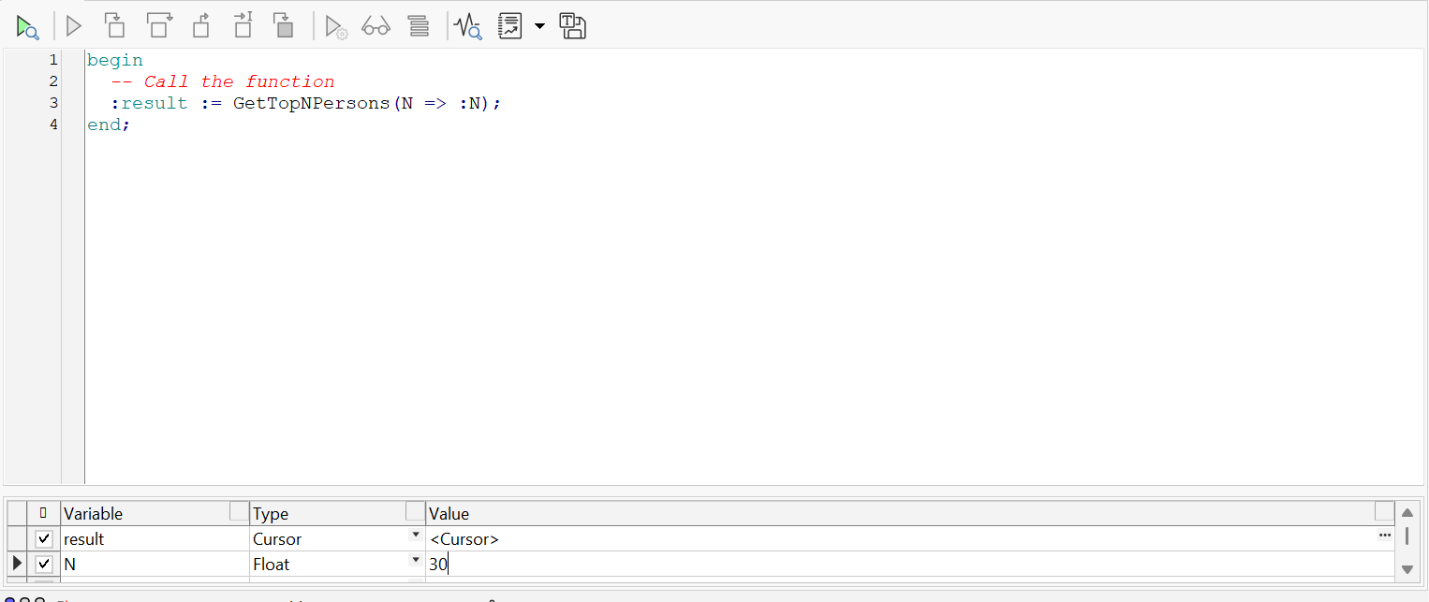
END IF;

END;

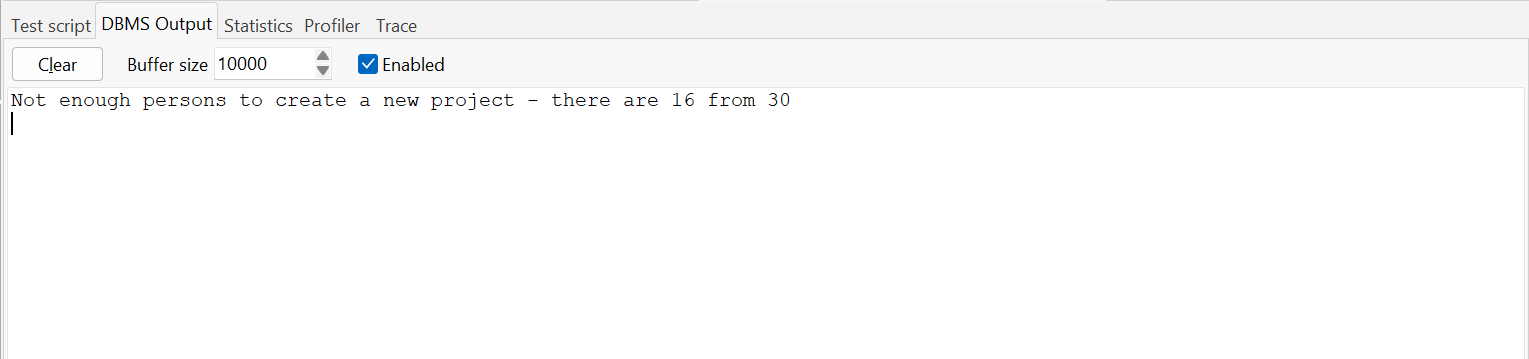
**תוצאות ההרצה**

הערה: כפי שכתבנו, הפונקציה גם מדפיסה למסך, אבל גם מחזירה קורסור. נוכל להדגים כאן את ההדפסות, אבל לא את החזרת הקורסור.

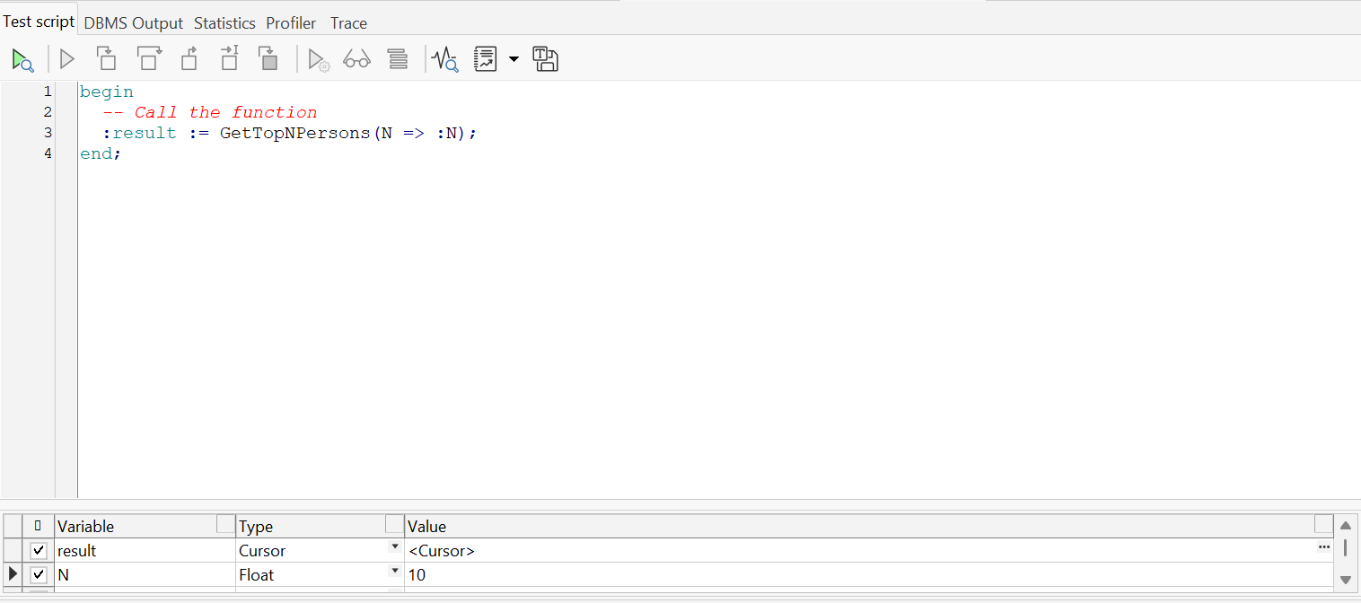
עבור ההרצה, נפתח חלון טסט שיוצר לנו אוטומטית קריאה לפונקציה הנ"ל.

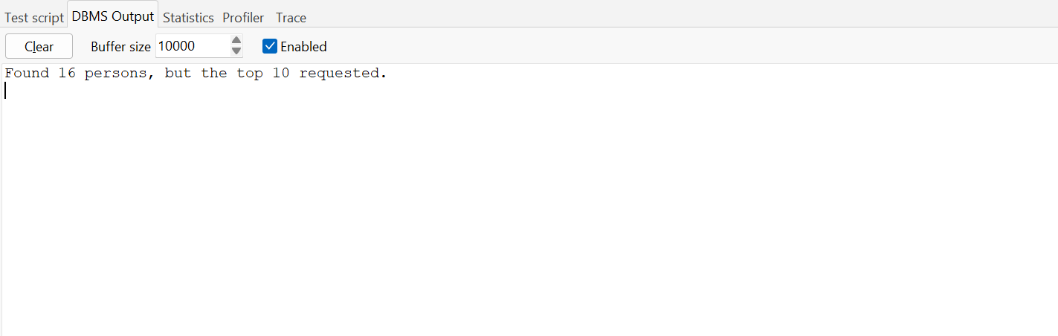
* נריץ את הפונקציה ונכניס ערך N=30 -

מסך הפלט יהיה -

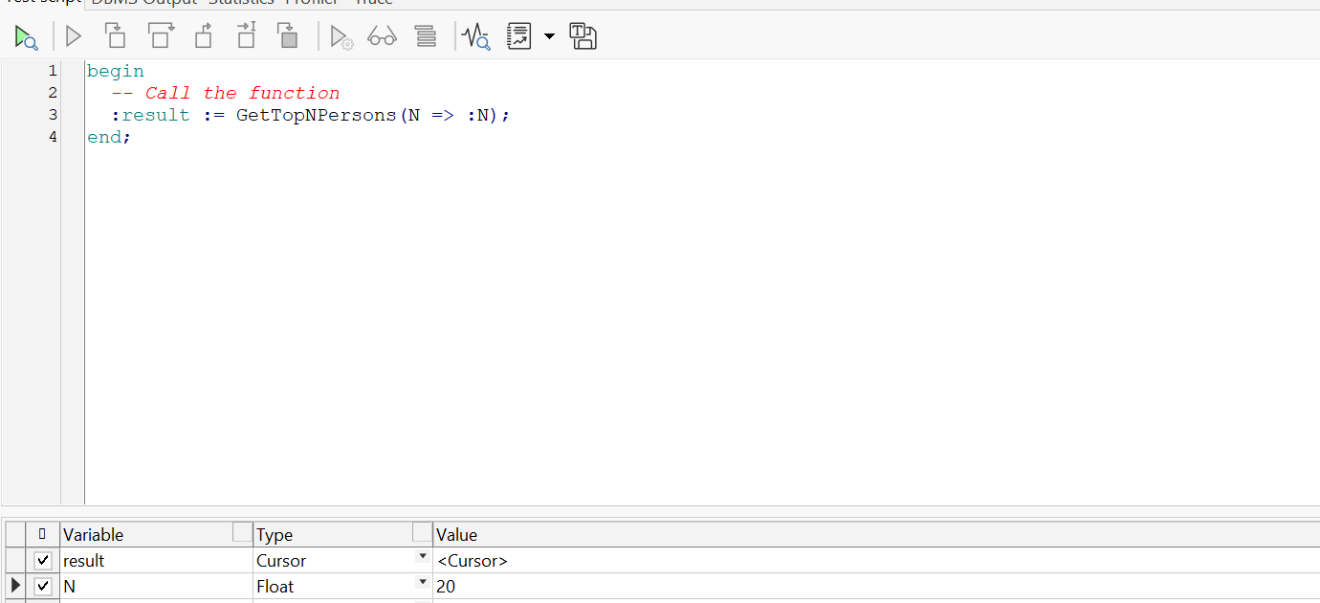


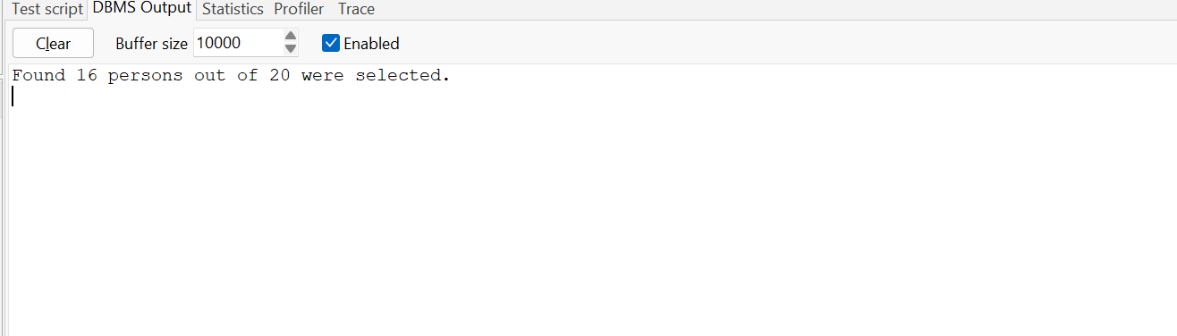
ניתן לראות שיש רק 16 אנשים העונים לקריטריון והפונקציה החזירה הודעה מתאימה.

* נריץ את הפונקציה ונכניס ערך N=10 -

מסך הפלט יהיה -

עכשיו הפונקציה החזירה הודעה שהיא מצאה 16 אנשים ובחרה את ה-10 הטובים.

* נריץ את הפונקציה ונכניס ערך N=20 -

מסך הפלט יהיה -

כאן אמנם אין מספיק אנשים, אבל יש למעלה מ-3/4 מהמבוקש ולכן הוא כן לקח את ה-16 שקיימים.

* + 1. **הפונקציה - GetUnitList**

**מטרה**

אנחנו רוצים רשימה של כל היחידות (UNIT) ועבור כל יחידה, את מספר הפרויקטים שהאנשים בה עובדים עליהם.

**תיאור**

הפונקציה מחזירה רף-קורסור כך שעבור כל יחידה הוא כולל את המאפיינים הבאים:

* מזהה היחידה (unitID)
* שם היחידה (unitName)
* מספר הפרויקטים שהאנשים ביחידה עובדים עליהם (projectCount)

**הקוד**

CREATE OR REPLACE FUNCTION GetUnitList

RETURN SYS\_REFCURSOR IS

v\_result\_cursor SYS\_REFCURSOR;

BEGIN

OPEN v\_result\_cursor FOR

SELECT u.unitID, u.unitName, COUNT(DISTINCT ip.projectID) as projectCount

FROM Unit u

LEFT JOIN Person p ON u.unitID = p.unitID

LEFT JOIN inProject ip ON p.personID = ip.personID

GROUP BY u.unitID, u.unitName

ORDER BY u.unitID;

RETURN v\_result\_cursor;

END;

**תוצאות ההרצה**

מכיוון שהפונקציה מחזירה קורסור, זה קצת קשה להריץ אותה בפני עצמה ולהראות את התוצאות.

נדגים את תוצאות ההרצה כאשר נראה את התוכנית הראשית שקוראת לפונקציה.

* 1. **תוכניות ראשיות**
     1. **התוכנית - openExcellentProject**

**מטרה**

אנחנו רוצים להרים פרויקט חדש של מצטיינים (כלומר, אנשים שהגיעו מיחידות מצטיינות).

**תיאור**

התוכנית קוראת לפונקציה **GetTopNPersons** שמחזירה קורסור עם תוצאות השאילתה - כלומר, האנשים שנבחרו לפרויקט, בהנחה שהפרויקט אכן נפתח. אחרת, NULL.

אח"כ, אם אכן יש מספיק אנשים, התכנית קוראת לפרוצדורה **CreateNewProjectAndAddPersons** עם הקורסור הנ"ל ושם הפרויקט החדש והפרוצדורה יוצרת פרויקט, מצרפת את האנשים ומדפיסה את פרטי האנשים ופרטי הפרויקט.

**הקוד**

DECLARE

v\_projectName VARCHAR2(100) := 'projectA';

v\_personCountNum NUMBER := 20;

v\_cur SYS\_REFCURSOR;

BEGIN

*-- Call function to get top N persons*

v\_cur := GetTopNPersons(v\_personCountNum);

IF v\_cur IS NOT NULL THEN

CreateNewProjectAndAddPersons(v\_cur, v\_projectName);

END IF;

EXCEPTION

WHEN OTHERS THEN

DBMS\_OUTPUT.PUT\_LINE('An error occurred: ' || SQLERRM);

END;

**תוצאות ההרצה**

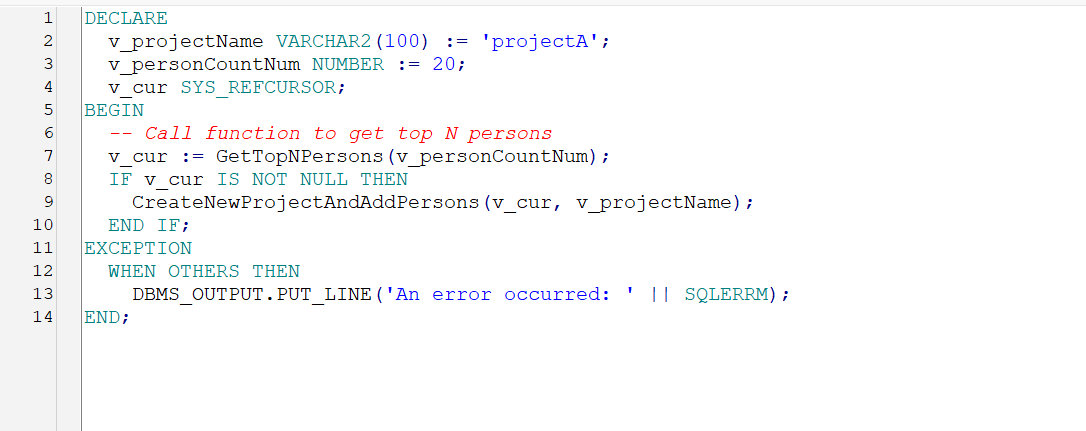
נריץ 2 מקרים.

* + יש מספיק אנשים עבור הפרויקט.

נפתח חלון TEST ונעתיק אליו את קוד התוכנית.

נריץ את התוכנית כאשר נשאיר את הערך 20 במשתנה **v\_personCountNum**.

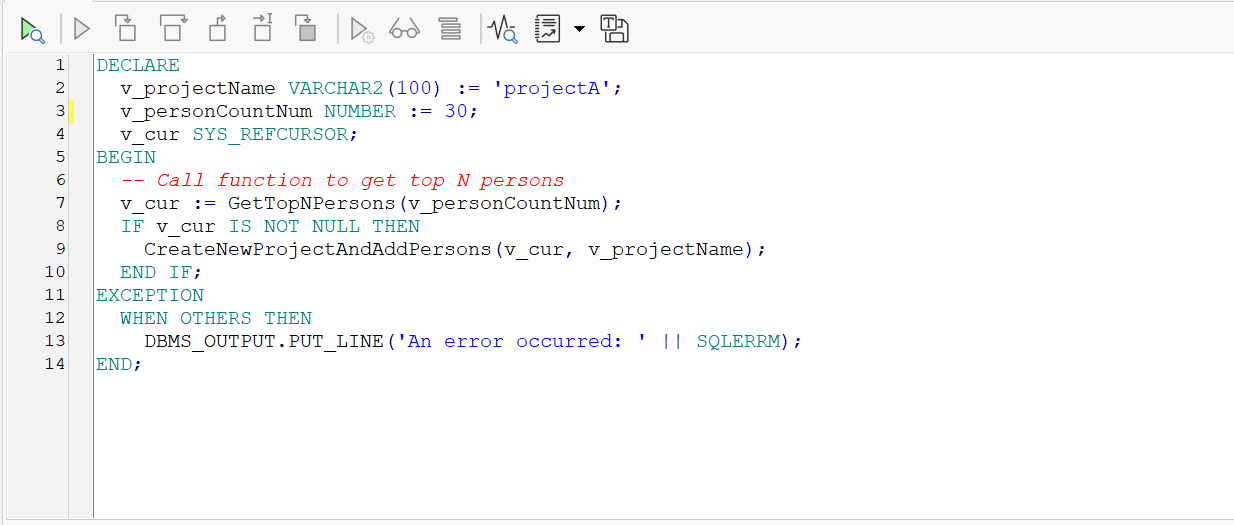
תמונה שמכילה טקסט, צילום מסך, גופן, מספר

התיאור נוצר באופן אוטומטיהפלט הוא -

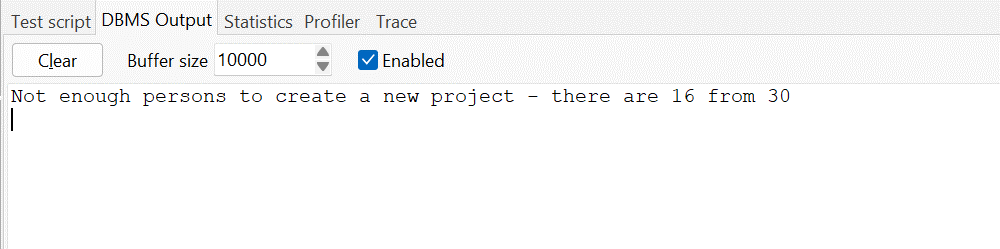
* + אין מספיק אנשים עבור הפרויקט.

נפתח חלון TEST ונעתיק אליו את קוד התוכנית.

נריץ את התוכנית כאשר נשאיר את הערך 30 במשתנה **v\_personCountNum**.



הפלט הוא -



* + 1. **התוכנית - AllocateResources\_Main**

**מטרה**

התוכנית מקצה משאבים לכל היחידות (UNIT) לפי מספר הפרויקטים שהאנשים בכל יחידה עובדים עליהם וסטטוס ההצטיינות של היחידה.

**תיאור**

התוכנית קוראת לפונקציה GetUnitList שמחזירה קורסור עם רשימת כל היחידות ומספר הפרויקטים של כל אחת מהן

לאחר מכן, היא עוברת על כל יחידה וקוראת לפרוצדורה AllocateResourcesForUnit עבור כל יחידה, כדי להקצות לה משאבים.

בנוסף, התוכנית גם מדפיסה דו"ח קטן עבור כל הקצאה (כמה משאבים הוקצו וכו').

**הקוד**

DECLARE

TYPE unit\_info\_rec IS RECORD (

unitID Unit.unitID%TYPE,

unitName Unit.unitName%TYPE,

projectCount NUMBER

);

v\_unit\_info unit\_info\_rec;

v\_unit\_cursor SYS\_REFCURSOR;

v\_resource\_type Resources.resourcesType%TYPE := 'low level';

v\_total\_units NUMBER := 0;

v\_processed\_units NUMBER := 0;

BEGIN

DBMS\_OUTPUT.PUT\_LINE('Starting resource allocation process' || CHR(10));

*-- Get the list of units*

v\_unit\_cursor := GetUnitList();

*-- Count total units*

SELECT COUNT(\*) INTO v\_total\_units FROM Unit;

*-- Process each unit*

LOOP

FETCH v\_unit\_cursor INTO v\_unit\_info;

EXIT WHEN v\_unit\_cursor%NOTFOUND;

v\_processed\_units := v\_processed\_units + 1;

DBMS\_OUTPUT.PUT\_LINE('Processing Unit ' || v\_unit\_info.unitID || ' - ' || v\_unit\_info.unitName);

DBMS\_OUTPUT.PUT\_LINE('Projects: ' || v\_unit\_info.projectCount);

*-- Allocate resources for the unit*

AllocateResourcesForUnit(v\_unit\_info.unitID, v\_resource\_type);

*-- Progress update*

DBMS\_OUTPUT.PUT\_LINE('Progress: ' || v\_processed\_units || ' out of ' || v\_total\_units || ' units processed');

DBMS\_OUTPUT.PUT\_LINE('--------------------' || CHR(10));

END LOOP;

CLOSE v\_unit\_cursor;

DBMS\_OUTPUT.PUT\_LINE('Resource allocation process completed');

EXCEPTION

WHEN OTHERS THEN

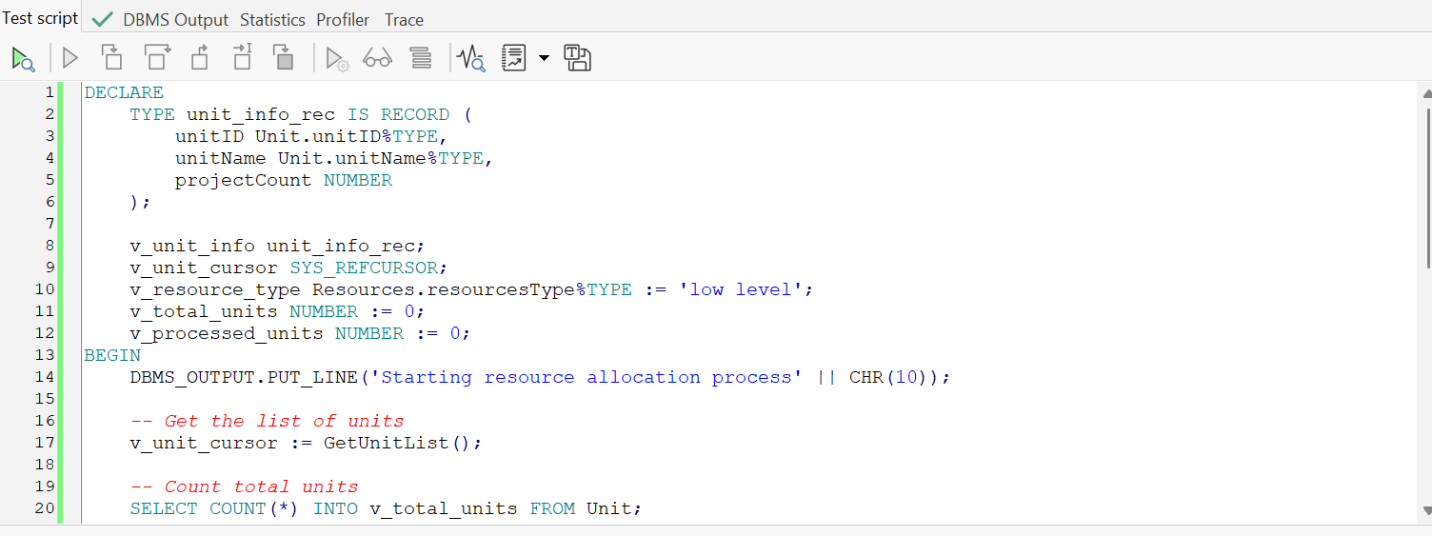
DBMS\_OUTPUT.PUT\_LINE('Error in MainProgram: ' || SQLERRM);

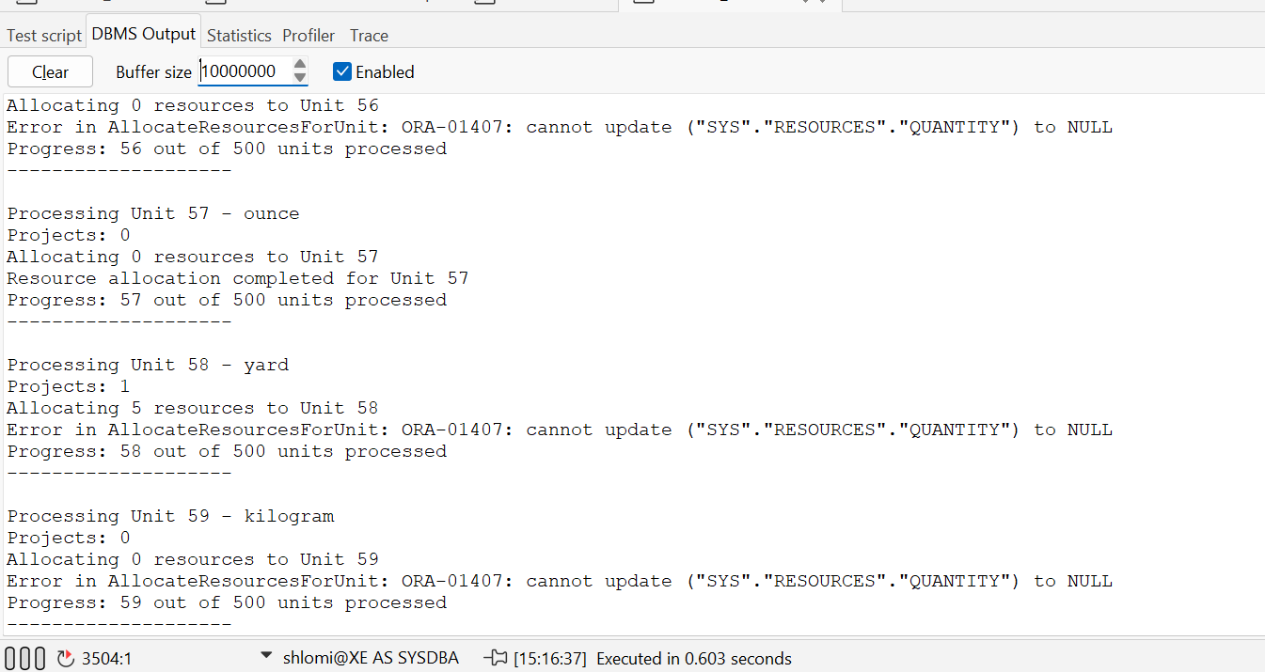
RAISE;

END;

**תוצאות ההרצה**

נריץ את התוכנית

הערה: מכיוון שיש המון הדפסות, אי אפשר להריץ ללא הגדלת החלון.

הפלט הוא -