



מינו"פ בסיסי נתונים – תשפ"ד

הצotta:

- ידידה בקורסזה: 332461854
- מאיר קרומבי: 214736688

על הארגון:

ברוכים הבאים למילון "אל הוטל לה פארם'אן" מלון 5 כוכבים מפואר בלב מזרח ירושם מקום פסטורלי בהחלט!

בקובץ זה נתאר את הליך בניית בסיסי הנתונים למחלקה ניהול העובדים במילון. בסיס הנתונים יכול את כל המידע על:

1. העובדים במילון – עובדים, מנהלי המחלקות, מג'יס'ים וכו'
2. מחלקות המילון – ניקיון, אוכל, פרסום, לוגיסטיקה, הנהלת חשבונות, הנהלה וכו'
3. אנשיים שמגישים מועמדות לעבוד באחת ממחלקות המילון.
4. משמרות – כמה משמרות כל עובד לקח ואיזה משמרות יש וכו'

שלב א – תיאור הארגון, מודל ERD, מודל DSD, סקריפטי SQL ואכלואס מידע:

הגדרת הישויות:

בבסיס נתונים זהה יכולול 7 ישותות כי הגדלנו ראש ואילו הן:

- **איש** - ישות אב הכללת תוכנות בסיסיות של הומוספיאנס שאנו חנו.

ישות זו מכילה את התוכנות הבאות:

1. **מספר זהות** – מפתח.
2. שם משפחה.
3. שם פרטי.
4. כתובות מגוריים.
5. עיר מגוריים.
6. אימיל
7. יום הולדת

- **עובד** – ישות המתארת את עובדים המלוון, מנהלים מנקיים ומה שביניהם

ישות זו ירושת מישות האיש את תוכנותיה, ובנוסף:

1. **מספר זהות** – ירושה מישות "איש" – מפתח ראשי (זר)
2. שכר שנתי.
3. תפקיד.
4. תאריך הצטרפות לצוות המלוון.

- **מעמד** – ישות המתארת את האנשים שהגישי מועמדות למשרה לאחט מחלקות המלוון.

עובדים אלה יכולים להגיש מועמדות למשרה בעבודה בלבד או על ידי המגייס שלהם (באם גוייסו על ידי אחד).

כל מועמד רשאי להגיש מועמדות למשרה אחת או יותר, בלבד או עם מגיסס ובלבד שלא ניתן ליותר ממשרה אחת למחלקה מסוימת!

1. **מספר זהות** - ירושה מישות "איש" – מהוות מפתח ראשי לטבלה זו.
2. [אופציוני] מספר הזהות של המגייס – תוכנה זו אפשרית ולכן ניתנת להיות בעלת ערך אילичי הואיל וכל מועמד רשאי להגיש את עצמו לעבוד.
3. תאריך התמודדות (נמצא בקשר "Willing_To_Work_A")

- **מנהל** – ישות המתארת את האנשים היושבים על הארגזים, אחראים על מחלקות ובעיקר פה



כדי לקבל 50% יותר משכורת מכולים.

ישות זו ירושת את תוכנותיה מישות העובד ובנוסף מקבלת מפתח זר של "מספר מחלקה".

- **מגייס** - ישות המתארת את העובדים האחראים על גיוס עובדים חדשים לצוות המלוון.

ישות זו ירושת את תוכנותיה מישות העובד.

- **מחלקה** - ישות המתארת את המחלקות השונות במלוון כגון מחלקת חשבונות ניקיון וכו'

לכל מחלקה מנהל אחד או יותר. (לפי הקשר שיוצג בתמונה בהמשך ☺)

תוכנות הישות:

1. **מספר מחלקה** - מפתח.
2. שם מחלקה.

- **শמרות** - ישות המתארת את כל המשמרות שעובד במלוון יכול לחת.

תוכנות הישות:

1. שעת התחלתה – מפתח.

2. שעת סיום.

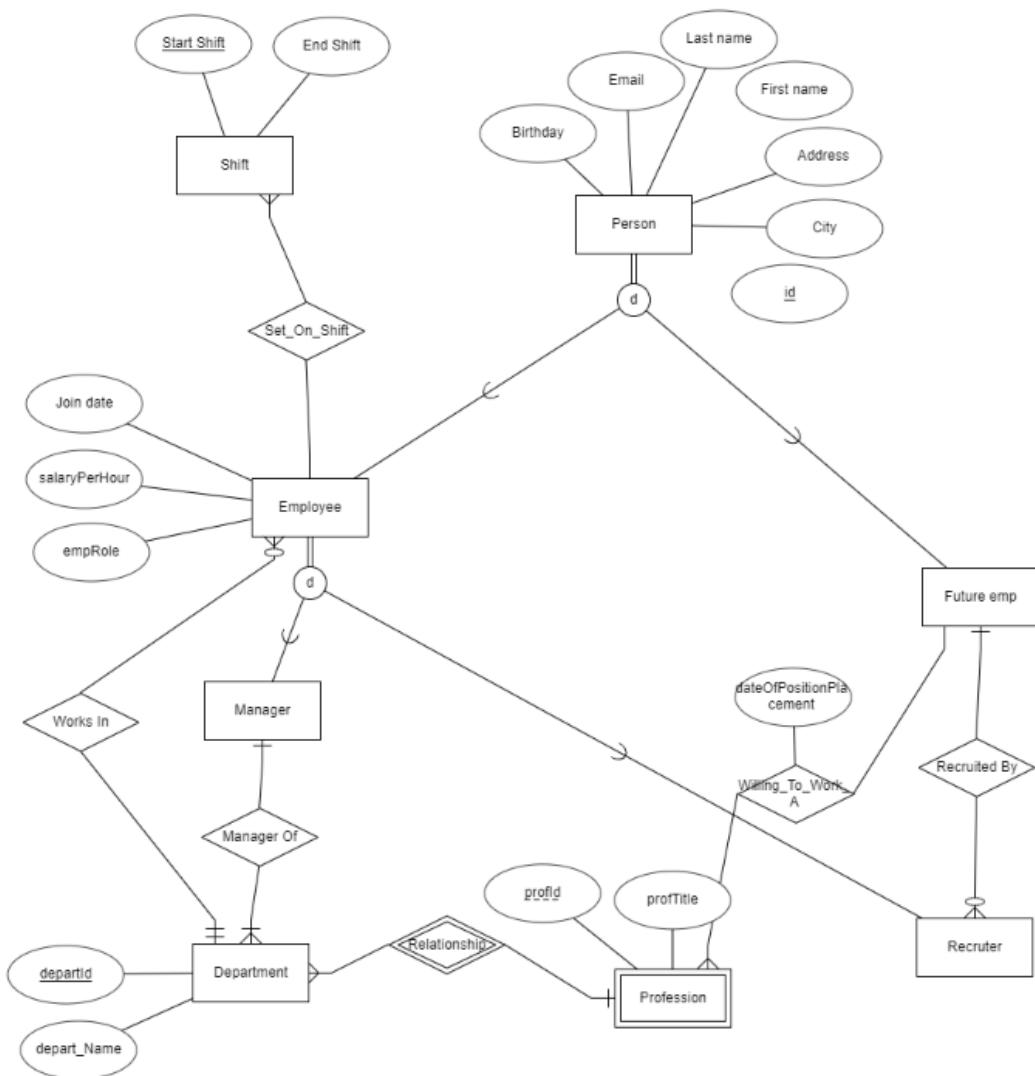
- **התמונות** – יישות המטאරת את תחומי העבודה של כל עובד במלון (כגון רואה חשבון ומצריה)

תכונות היחסות:

1. מס' תפקיד – מפתח חלש לעומת "ישות מחלקה" (אפשר לראות בדיאגרמת (ERD)).
2. תיאור תפקיד ..

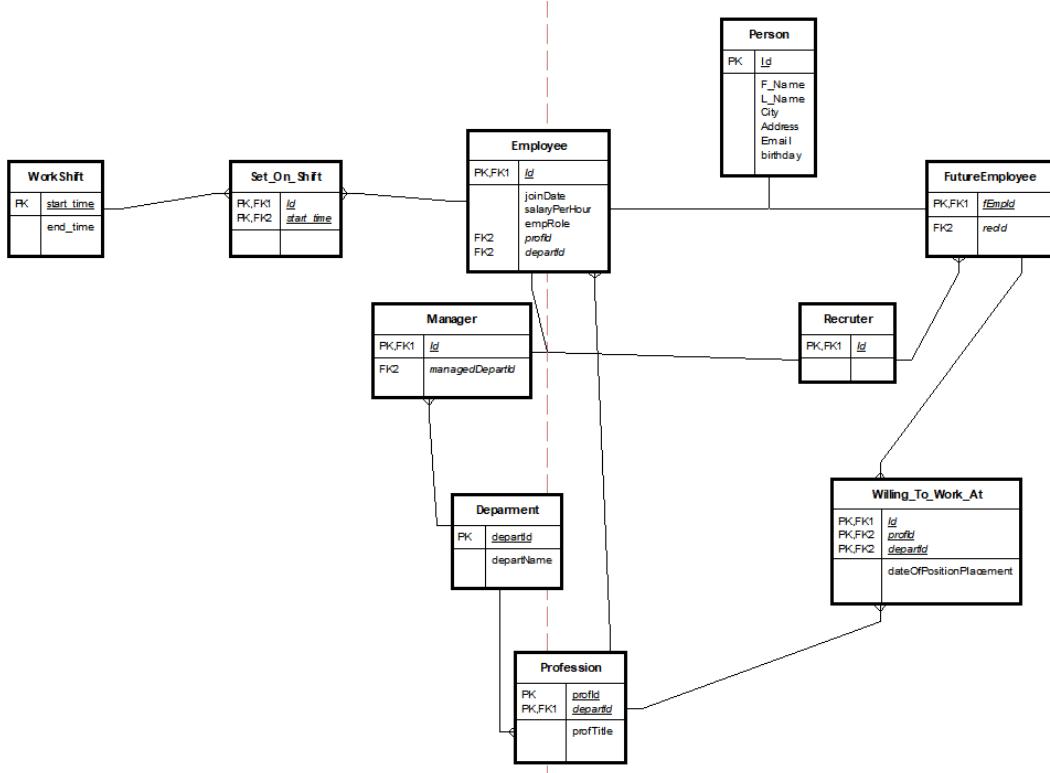
דיאגרמת ERD

הDİאגרמה נעשתה בתוכנת DDS lite – כמו שהמורה אמר לנו. זו בעצם דיאגרמה שמתארת בצורה ויזואלית את כל מה שהוסבר על היחסות בחלק הקודם.



דיאגרמת DSD

תיאור הטבלאות שאנו ניצור במערכת בסיס הנתונים שלנו – ניתן לראות את המפתחות הראשיים בכל טבלה וכן כיצד ניתן להבחין כי חלק מהקשרים מטבלת ERD אינם מתוארים בטבלאות בפני עצם אלא מתוארים כמפתחות זרים בטבלאות אחרות.



שינוי נוסף שלא ניתן לעשותו בתוכנה היא קביעת מספר תכונות כמפתח ראשי לטבלה.

במקרה דן רצינו ליצור את טבלת FutureEmployee כך שהמפתח הראשי לטבלה זו תהיה ת"ז העובד העתידי (fEmpId) וגם מספר המחלקה שבה הוא ירצה לעבוד (departId) – שכן עובד לא יכול מועמדות לאותו התפקיד יותר מפעם אחת. אך ללא הצלחה נאלצנו להשאיר זאת לשלב כתיבת הסקריפטים.

נרטול טבלאות – NF3

כל הטעיות שלנו מנו רמלות ל-NF3, נרטול טבלאות לצורה הזאת הוא מאוד חשוב מהסיבות הבאות:

1. מבטל יתרות על ידי פירוק טבלאות לחידות אטומיות קטנות יותר.
2. גורם לשיפור שלימות הנתונים על ידי הפחתת הסיכון לחריגות.
3. מקל על מדריגות וגמישות של מסד הנתונים על ידי מתן בסיס איתן לשינויים.
4. משפר את עקביות הנתונים על ידי אחסון נתונים באופן לא מיותר.

LOSECOM –

נו רמליזציה של טבלאות ל – NF3 משפרת את איזות הנתונים, מפחיתה יתרות וחריגות, מבטיחה שלמות ועקביות נתונים, מפשטת את התחזקה ותומכת במדריגות וגבישות של מסד הנתונים.

יצירות טבלאות –

בשלב זה נתאר את הטעיות שיצרנו עבור בסיס הנתונים שלנו. עבור כל טבלה אנו נתאר ונביא את תיאור הטבלה כפקודת create table ב-SQL, את ההליך בה הכנסו את הנתונים דוגמיה קטנה של הנתונים.

יאלה מתחילה –

איש איש מה אתה מרגיש – טבלת האיש Person:

טבלת האיש (Person בלו"ז) מכילה בתוכה אוסף של כל האנשים שקיים במסד הנתונים שלנו.

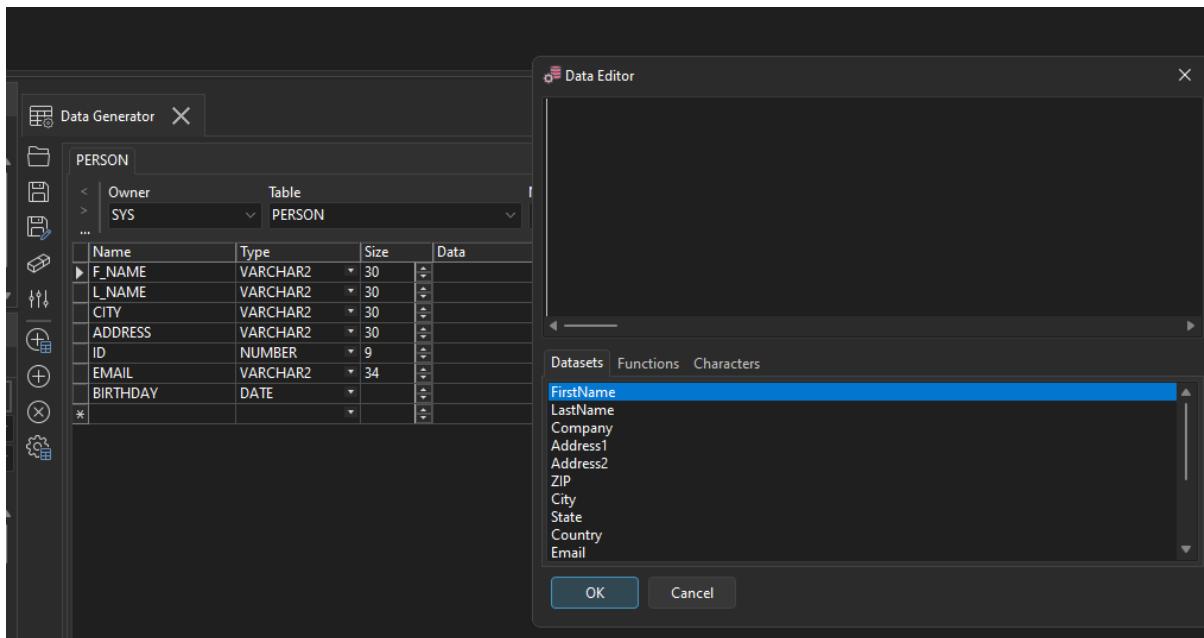
אוסף האנשים כולל בין היתר את העובדים, המועמדים וכאלה שהם סתם אנשים (סתם יחסית לתפקידים – אף אדם אין "סתמי") ולכן היא מהווה ישות אבעור ישויות בן שירותם ממנה את תוכנותיה.

את החלוקת וישיות הבן הירושות נתאר בהמשך הטעיות, כתעת נחזור לישות האב –

כפי שניתן לראות ישות ה-"איש" מכילה בתוכה תוכנות בסיסיות של כל אדם, כמו שם משפחתו, העיר והכתובת בה מתגורר, תאריך יום הולדתו וכתובת דוא"ר האלקטרונית (כн, זה המצביע כיום שהמייל מתאר את האדםנו שווין).

```
CREATE TABLE Person (
    F_Name      VARCHAR2(30) NOT NULL,
    L_Name      VARCHAR2(30) NOT NULL,
    City        VARCHAR2(30) NOT NULL,
    Address     VARCHAR2(30) NOT NULL,
    Id          NUMBER(9) NOT NULL,
    Email       VARCHAR2(34) NOT NULL,
    birthday    DATE NOT NULL,
    CONSTRAINT pk_Person PRIMARY KEY (Id))
/
```

از נעבר לתוכלו – איך הכנסנו את הנתונים? ובכן, ל-PLSQL יש מחולל נתונים מותםע בתוכנה (Data Generator) ובו קיימים מערכិ נתונים מוגדרים מראש שנפוצים כגון: שם, שם משפחה, כתובות וכו' –



דבר אחד שהוא חסר לנו היה מאגר מידע של תאריכים עבור תאריכי לידה של כל אדם מקובצת הישיות "איש" – דבר זה לא היה קיים במערכת ולכן נאלצנו כמו כל מפתח טוב לרדת לעמקי השאלה, אל הלא נודע, אל המעמיקים – "אל התיעוד (Documentation)" (אל מלשון לכיוון ולא משולם ישות שמיונית אלהיה צאת או אחרת, למרות שכמה מפתחים מתיחסים אליו כאחד זהה).

از הנה אנחנו בעיצומו של צאת יומן זיכרון יושבים ושוברים את הראש במאבק למצוא את הטריק, את הדרך בה נוכל למש את זה.
ואז ראיינו את האור –

There are also some example user-defined data sets available

- Components.Code – General article items: Article code
- Components.Description – Article description (Computer parts)
- Components.Price – Article price
- Elements.Name – Chemical elements (name)
- Elements.Symbol – Chemical elements (symbol)

These data sets can be found in the DataGenerator\ UserData directory as elements.txt and components.txt. You can add your own sets if you want. Simply add a comma-separated file where the first line holds the description between square brackets. You can use data from your file by specifying filename.description like the two examples.

All the functions and data mentioned above can be added together, for example: Random(10..99) + '.' + [A(4)]

The + is optional, but there should at least be a space as separator.

כפי שניתן לראות בחלק השני של הקטע זהה (עמוד 196) – נאמר בו שם נרצה להוסיף מאגרי מידע למחולל הנתונים אז עליינו ליצור קובץ עם הערכים כאשר הם מופרדים על ידי פסיק. ובנוסף, בראש הקובץ והעמודה עליינו לחת לתוכן לערכים שבאותו העמודה מה שי יכול לתת לנו את האופציה ליצור כמה עמודות באותו הקובץ – אבל אנחנו מעוניינים רק באחד מהם אז הכל בסידר.

از עכשו נשאל השאלה – היכיז ניצור מאגר מידע שכזה?
אני שמח ששאלתכם כי לנו יש תשובה לזה והוא כולל את החית מוחמד אהובה עליינו –



לא הפיתון הזה יא גאון הפיתון השני –



או שכו"ח בדיק מה שרצינו.

אך ככה מה שאנו עושים הוא ניצור סקריפט בפייתון שייגנרט (מלשון Generate, המילה "יחולל"
לא נשמעת לי כל כך טובה) לנו מלא תאריכי לידי בין השנים 1974 עד 2004
הסקריפט מייצר קובץ csv שבו נתונים ערכים רנדומליים של תאריכים כפי שצינו.
בסוף הוא ישמור את הערכים בתבנית –

```
TO_DATE('date')
```

כך שכאשר המוחלט הנתונים יקרא את זה הוא יפרש את זה כערך מסוג DATE ולא מסוג NUMBER
או VARCHAR2.

שנראת את הסקריפט? נו ברור אחרת לא תנתנו לנו ציון לא ככה? –

```
from datetime import datetime, timedelta
import random

start_date = datetime(1974, 1, 1)
end_date = datetime(2004, 12, 31)

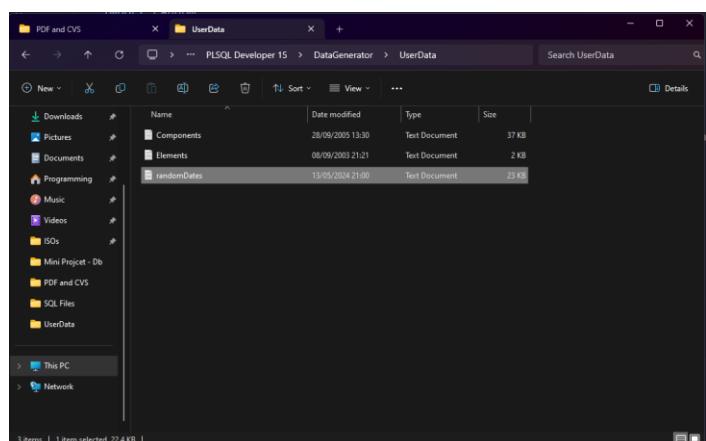
dates = set()
while len(dates) < 1000:
    random_date = start_date + timedelta(days=random.randint(0, (end_date - start_date).days))
    dates.add(random_date)

dates_list = list(dates)

output_file = 'PDF and CVS\generatedRandomBirthdayDates.txt'
with open(output_file, 'w') as f:
    f.write('[RANDOM]\n')
    for date in dates_list:
        formatted_date = date.strftime('%Y-%m-%d')
        f.write(f"TO_DATE('{formatted_date}')\n")
```

אrrorררר התענוג כשאתה לא צריך לעשות הרבה עבודה ... אמריקה
 אז כפי שניתן לראות הוא מייצר את הקובץ עם הערך [RANDOM] בתחילת הקובץ המהווה שם
 שרירותי עבור העמודה של הערכים.
 כתת כל מה שנותר לנו הוא לcliffe תיקייה הראשית של המחשב – אל תיקייה Program Files לחפש
 את התקייה של Plsql Developer ולהיכנס אל תיקייה UserData ולהכנס שם את הקובץ
 המג'נט –

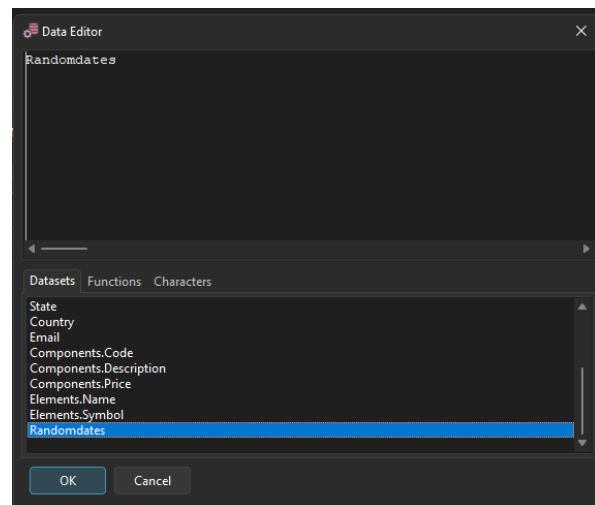
בינתן לשנות את השם על מנת לקבל
 תיאור יותר מינימלי במחולל הנתונים על
 הערכים שהקובץ זהה מכיל.



להפעיל מחדש את PLSQL והפלא ופלא –

כעת ניתן לראות את הקובץ במחולל הנתונים ולייצור שורות של אנשים עם שימוש בתאריכים האלו.

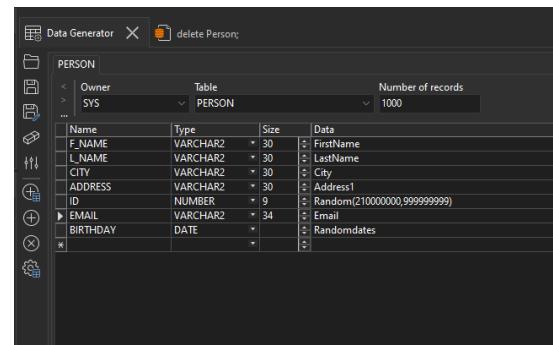
נכון שמניב? חכו להמשך.



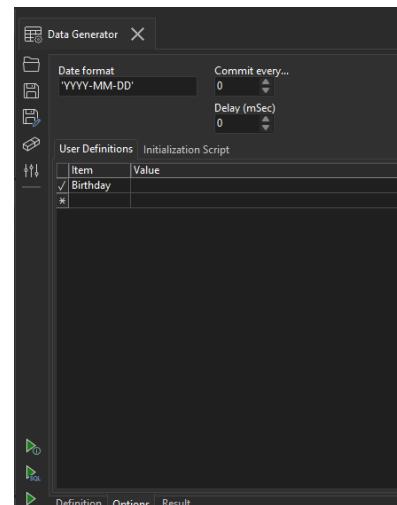
כעת סוף סוף אפשר לייצר את הערכים עבור הטבלה Person בקלי קלות –

בשורת הת"ז (pi) נבקש שייצר לנו מספר אקראי בין
999999999 ל- 210000000

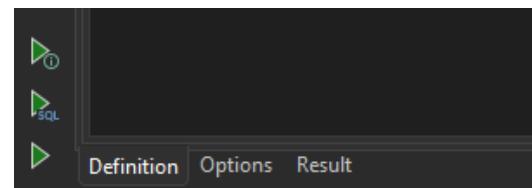
نبחר את המאגרי המידע הנחוצים לנו לכל אחת מהשורות ונבקש מהמחולל הייר שיחול עבוריינו סך של 1000 שורות



לאחר מכן נלחץ על הכption Options, נכתוב את הפורמט – YYYY-MM-DD' ונוסיף את השורה birthday כפי שמיוצג פה על מנת להגיד למוחולל הנתונים שהנתונים במקומות של birthday הם מהפורט שרשמנו לעיל.



נלחץ על הכפתור הירוק העליון כדי לבדוק איך המידע
יראה (לא להילחץ מזה שבעמודה של התאריך לידה
המידע לא נראה כמו DATE זה השתנה במהלך
הגינורוט של המידע) ולאחר מכן על הכפתור האחרון
כדי להריץ את הכל



The screenshot shows a Data Generator interface with a query window containing:

```
select *
from Person;
```

Below the query is a large table with 19 rows of generated data:

	F_NAME	L_NAME	CITY	ADDRESS	ID	EMAIL	BIRTHDAY
1	Nicholas	Gill	Austin	73 Liu Ave	422871835	n.gill@linksys.com	08/09/1986
2	Jonny	Broderick	Wavre	95 Hatosy Ave	736044763	jonny@pearlawgroup.be	26/06/1999
3	King	Donovan	St Leonards	96 Wopat Drive	330720212	king.donovan@newmedia.au	22/03/1999
4	Tcheky	Fehr	Bellevue	62 Rhys Road	831833319	tcheky.fehr@lydiantrust.com	14/01/2000
5	Beth	Field	Cheshire	93 Witt Road	286338422	beth.field@calence.uk	20/12/1975
6	Ali	Guzman	Huntington	53rd Street	357206369	ali.guzman@fordmotor.com	22/09/1993
7	Miles	Apple	Hercules	51st Street	528538413	miles.apple@solipsys.com	27/02/2004
8	Alec	Weaving	Cesena	26 Fisher Drive	952491494	a.weaving@abatix.it	06/03/1995
9	Louise	Page	Warszawa	52nd Street	294544412	l.page@trx.pl	19/08/1976
10	Alicia	Wilson	Harahan	49 Percy	944661688	alicia.wilson@maverick.com	29/05/1985
11	Maureen	Elizabeth	Berkshire	634 Cooper Ave	354095319	maureen.elizabeth@tilsonlandscape.	14/02/1997
12	Sigourney	Whitman	Tustin	71st Street	744938085	sigourney.whitman@sony.com	02/04/1989
13	Andrea	Chaykin	Perth	89 South Jordan Ave	730307131	andrea@viacom.au	10/10/2002
14	Bobby	Presley	Eisenhüttenstadt	56 Francis Street	941628619	bobby.presley@unilever.de	13/03/1990
15	Murray	Lynskey	Changwon-si	92 Alleroed	933502920	murray@kelmooreinvestment.com	23/12/1992
16	Hex	Rebhorn	Chaam	52 Morgan Drive	297711531	h.rebhorn@ams.nl	24/01/1995
17	Rosario	Postlethwaite	Enschede	33rd Street	664772674	r.postlethwaite@telepoint.nl	25/03/2002
18	Chris	Epps	Herford	18 Cocker Ave	861139535	chris.epps@learningvoyage.de	18/11/1980
19	Gino	Beatty	Vista	80 Robbie Drive	808746130	g.beatty@news.com	25/01/1980

אואויקה! יש לנו את זה חברים יקרים הצלחנו לג'נרט את המידע דרך מחולל הנתונים.

cut נ עבור לatable הבא.

מחלקות ולא בקטע של OOP – טבלת המחלקות והתפקידים, **Department :Profession**

כמו שלכל סיר יש מכסה – לכל מנהל יש משועבד ולכל משועבד יש מהו שהוא משתעבד אליו.

אנחנו ניצור כעת טבלה של מחלקות ותפקידים לכל מחלקה – את הקבצים האלה ימראנו בעצמינו אך בבקשת מחיאות כפיים כי יש פה ליטרלי 1500 פלאס שורות שנעשו באופן ידני!

אך ככה –

כפי שזכרנו לכם או שאתם רואים את זה כעת מול עיניכם – לכל מחלקה יש עובדים השיכים למחלקה הזאת ולכן הקבוצה של התפקידים היא ישות חלה יחסית לקבוצה של המחלקות, זה מرتبط בכר של קבוצה של התפקידים יש מפתח ראשי שמורכב מהມפתח שלו ושל הטבלה של המחלקות.

```

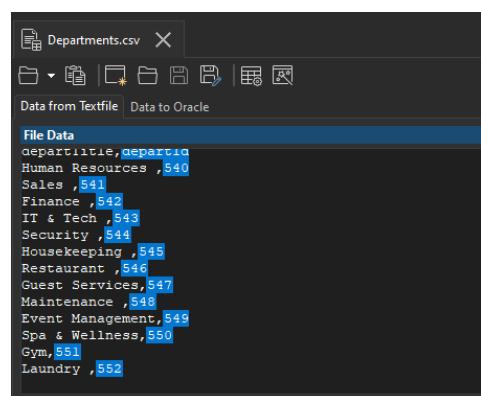
CREATE TABLE Department (
    departId      NUMBER(3) NOT NULL,
    departTitle   VARCHAR2(30) NOT NULL,
    CONSTRAINT pk_Department PRIMARY KEY (departId)
)

CREATE TABLE Profession (
    profId       NUMBER(9) NOT NULL,
    departId     NUMBER(3) NOT NULL,
    profTitle    VARCHAR2(30) NOT NULL,
    CONSTRAINT pk_Profession PRIMARY KEY (profId,departId),
    CONSTRAINT fk_Profession FOREIGN KEY (departId)
        REFERENCES Department (departId)
    ON DELETE CASCADE
)
```

לאחר שיצרנו את הטבלאות הבא ונכנסו לתוכה את הערכים שלה – בריפו תוכלנו למצוא את הקבצי csv שמכילים את המחלקות ואת התפקידים (Departments.csv) וprofession.csv).

נשתמש בשיטה השנייה של הכנסת נתונים והיא על ידי שימוש בקבצי csv ושימוש בכלי Text Importer שיש לנו באנדרואיד.

בחר את הקובץ Departments.csv על ידי לחיצה על הכפתור של התקינה.

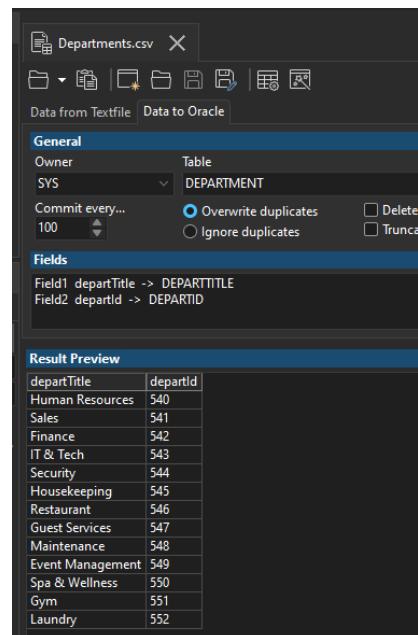


לאחר מכן נבחר את הטבלה שנרצה להכניס אליה את הנתונים – במקורה דן Department

שיםו לב שהתוכנה מזהה את הקשר בין העמודות של הטבלה לעמודות של הקובץ היות ונתנו להם את אותם השמות.

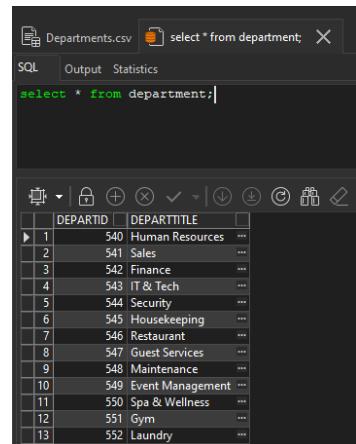
למטה ניתן לראות איך הטבלה תראה – אל דאגה יש לנו טבלאות יותר מסיביות

נלחץ על הכפתור **Import** ונייבא את המידע

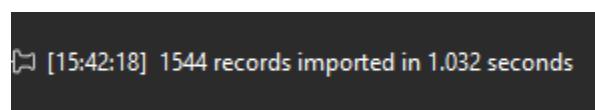


והנה לנו המידע שלנו

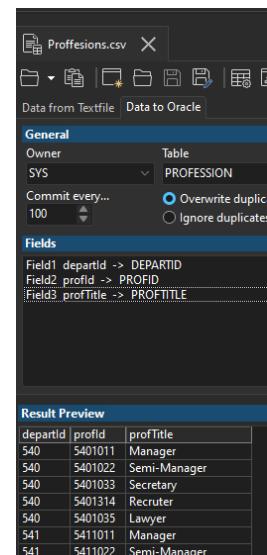
וכעת ניתן להמשיך להלאה אל התפקידים של כל מחלקה הקובץ professions.csv גם הוא ניתן לייבוא אל תוך בסיס הנתונים ונעשה זאת זה באותה הדרך



אם פה נבחר את הקובץ המקורי ואת הטבלת יעד שלנו ניתן לראות גם פה דוגמיה מתוך כל הנתונים המקוריים
נלחץ על יבוא ונ��וויה שהמחשב לא יקרוס (שוב) כתוצאה לכך



וכפי שניתן לראות הכנסנו 1544 שורות של נתונים.
כעת נ└ר לשעבד לנו כמה אנשים.



מוכר במדונלדס עוקץ המבורגר, ברמן בבר מבריח בקבוק – מה מבריחים המנקים? בקבוק אקונומיקה? – טבלת העובדים (Employee):

כעת ניצור את טבלת העובדים –

כידוע לכמ כל עובד מכיל בתוכו מפתח זר
מישות האב ("איש") – המספר זהות
שמתפקד כמספר ראשי בסכמתה זו

כמו כן ישות העובד מכילה בתוכו 2 מפתחות
זרים נוספות – מספר מחלוקת ומספר התפקיד
שהוא עובד בה במלון.

```
CREATE TABLE Employee (
    Id          NUMBER(9) NOT NULL,
    joinDate    DATE NOT NULL,
    salaryPerHour NUMBER(5,2) NOT NULL,
    profId      NUMBER(3) NOT NULL,
    departId    NUMBER(3) NOT NULL,
    CONSTRAINT pk_Employee PRIMARY KEY (Id),
    CONSTRAINT fk_Employee2 FOREIGN KEY (Id)
        REFERENCES Person (Id),
    CONSTRAINT fk_Employee FOREIGN KEY (profId,departId)
        REFERENCES Profession (profId,departId)
        ON DELETE CASCADE)
    /
```

לכן על מנת שניצור בסיס נתונים איקוטי ומיתתי علينا לוודא שהערכים שהעובדים מקבלים במפתחות זרים – הם בהתאם לערכים שבאמת קיימים בסיס הנתונים שלנו ולא כל מיני ערכים אפשריים אין ואף שום דבר. לכן علينا לוודא שהם באמת מקבלים ערכים אמיתיים והגיוניים.

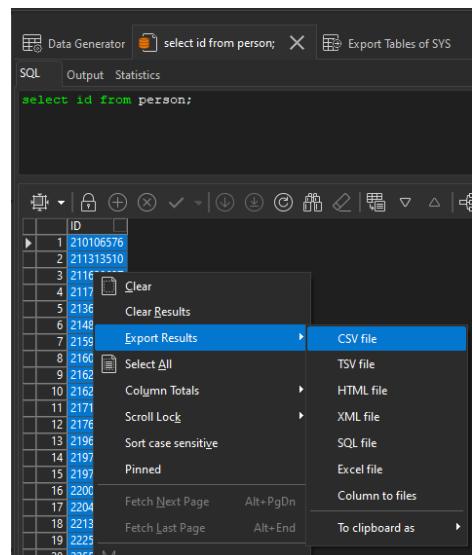
איך נעשה את זה? לשם כך נדרש דבר ראשון לקבל מאגר מידע שמכיל את כל מספרי הזהות שכבר רשומים במערכת.

הנה איך נעשה את זה, נפתח את PLSQL ונרשם את השאלה שתנתן לנו את כל האנשים שקיימים במערכת –

עַל מִנְתָּ לְקַבֵּל אֶת כָּל הַתוֹצָאָת מִבְּסֵיס הַנְּתָנוֹנִים תַּלְחִצּוּ עַל הַכְּפֹטוֹר הַזָּה
שְׁמוֹפִיעַ בְּשׂוֹרָה מֵעַל פָּאֵל הַתוֹצָאָת.

ולאחר מכן נלחץ על העמודה ID וזה יסמן את כל
שורות התוצאה של השאלה.

לחיצה ימנית על השורות המסומנות ולאחר מכן על
הקובץ המבוקש, עם אפשרות של השם שנבחר
שהמערכת תבקש מאייתנו כמו כן גם המיקום בה
נרצה לשמר את הקובץ נקבל את הקובץ המינוחל



עכשו כשים לנו את הקובץ בוואו נחלק את הערכים שלה לכמה חלקים, לחלק אחד שיצין את העובדים שכבר קיימים במערכת בתור עובדים, אנשים שעומדים להגיש מועמדות עם מגיס ואנשים שעומדים להגיש מועמדות ללא נציג.

זהו נוכל ליצור מידע שמתואם עם המזיאות הקיימת.
איך נעשה זאת זה? נשתמש באתר mockaroo הידע –

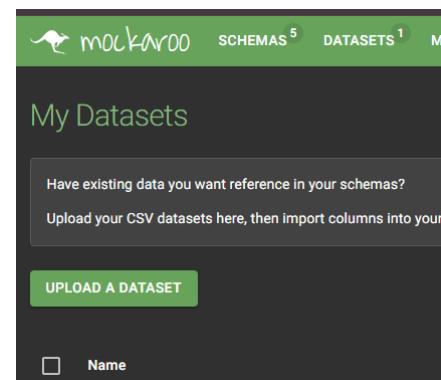
Employee Id	Future Employee Id - No Recruter	Future Employee Id - With Recruter
808746130	210106576	436496468
809382450	211313510	436786942
810071746	211639627	437083372
810506277	211718083	440216709
812680196	213605497	440449299
813140191	214805905	442616999
813151024	215933752	443593292
813411795	216067212	445043943
813748082	216236641	445127724
815027438	216253166	445302856
816369888	217184131	445897601
816892012	217692660	446981230
819334076	219663600	449111090
820536677	219710512	449115974
823527256	219770648	449211375
826951426	220072486	450013313
827268387	220479549	450167307
827976185	221393039	450928967
828720845	222558106	451904163

נחלק את הערכים שקיימים לנו מבסיס הנתונים בצורה הבאה –

- מספר זיהות הראשונים ילווה לאנשים שכבר עובדים במלון.
 - הבאים הם יהיו האנשים שמουמדים לעבודת במלון והגינו ללא עזרה של מגיס.
 - ומי שנשאר יהיו המועמדים שהגינו לעבודה עם עזרת המגייס
- ניתן לזה שם חדש ונשמר את זה כקובץ csv וcutet בלבד לאתר mockoo –

כפי שהיה לנו עם מחולל הנתונים גם פה נדרש ליצור מאגר מידע על מנת שנשאב ממנו את הערכים עבור השדות הרצויות – במקרה דנן אנו רוצים שהשדה id בשדות העובדים תהיה מתואמת עם השדה id שבדוחת האנשים.

נבחר datasets ונעלה מאגר מידע על ידי לחיצה על upload dataset



נעלה לשם את הקובץ וניתן למאגר הזה שם – Employee ids

כעת נודא שיש לנו מאגר מידע של התפקידים והמחלקות שלהם – נשתמש בקובץ `Profession.csv` שהשתמשנו בו קודם.

נעלה גם אותו לאתר [mockaroo.com](#) על מנת שתהיה לנו גישה אל המידע שבפניהם.

וכעת אנחנו מוכנים ליצור את השורות העובדים שלנו –

Field Name	Type	Options
<code>id</code>	Dataset Column	Person Ids Employee Id sequential blank: 0 % Σ X
<code>departId</code>	Dataset Column	Professions departId sequential blank: 0 % Σ X
<code>profId</code>	Dataset Column	Professions profId sequential blank: 0 % Σ X
<code>joinDate</code>	Datetime	01/04/2019 to 05/08/2023 format: yyyy-mm-dd blank: 0 % Σ X
<code>salaryPerHour</code>	Number	min: 28.78 max: 82.87 decimals: 0 blank: 0 % Σ X

+ ADD ANOTHER FIELD GENERATE FIELDS USING AI...

Rows: 997 Format: CSV Line Ending: Unix (LF) Include: header BOM

כפי שניתו לראותו אנחנו משתמשים בערכים שנמצאים בקובציים שהעלנו בתור מאגרי מידע (`ids` Professions Person) ואנחנו משתמשים בעמודות מסוימות בכל אחד מהקובציים האלה (`id` Employee ו`co`) ובاهדרות מסוימת למאגרי מידע קיימים כמו טווח תאריכים עבור תאריך הכניסה לעבודה (`joinDate`) ומספר מינימלי ומסקימלי עבור המשכורת (`salaryPerHour`) כਮובן שນבקש גם 2 ספרות לאחר המספר השלם. ובנוסף על ידי בחירה בבחירה `sequential` אנחנו מבקשים ממכלול הנתונים ליצור את הערכים כך שייעבור שורה אחר שורה ולא בצורה רנדומלית – היות ונרצה שעבודד יהיה מתואר פעמיים אחת בלבד (ובנוסף ביקשנו רק 997 ערכים להיות קיימים לנו רק 997 עובדים במאגר המידע).

ולאחר מכן נבקש ממנו שיוריד לנו קובץ SQL שמכיל בתוכו את פקודות `insert into` על מנת שנכניס את המידע לבסיס הנתונים שלנו –

נבחר את המאגר מידע שיצרנו קודם בחלוקת של Append Dataset נרשם את השם
של הטבלה שלנו ונலחץ על תוך SQL –

ניקח את הקובץ שירד למחשב שלנו ונזירוק אותו אל תוך SQL –

```
insert into Employee (id, departId, profId, joinDate, salaryPerHour) values (210522947, 540, 5401011, TO_DATE('2019-01-3
insert into Employee (id, departId, profId, joinDate, salaryPerHour) values (210929135, 540, 5401022, TO_DATE('2022-10-3
insert into Employee (id, departId, profId, joinDate, salaryPerHour) values (211107584, 540, 5401033, TO_DATE('2020-03-2
insert into Employee (id, departId, profId, joinDate, salaryPerHour) values (211317837, 540, 5401314, TO_DATE('2021-05-3
insert into Employee (id, departId, profId, joinDate, salaryPerHour) values (211540296, 540, 5401035, TO_DATE('2019-07-0
insert into Employee (id, departId, profId, joinDate, salaryPerHour) values (211553724, 541, 5411011, TO_DATE('2021-07-2
insert into Employee (id, departId, profId, joinDate, salaryPerHour) values (211716295, 541, 5411022, TO_DATE('2023-01-0
insert into Employee (id, departId, profId, joinDate, salaryPerHour) values (211773626, 541, 5411033, TO_DATE('2019-03-0
insert into Employee (id, departId, profId, joinDate, salaryPerHour) values (211972561, 541, 5411334, TO_DATE('2022-03-1
insert into Employee (id, departId, profId, joinDate, salaryPerHour) values (212356354, 541, 5411335, TO_DATE('2022-04-1
insert into Employee (id, departId, profId, joinDate, salaryPerHour) values (212561034, 541, 5411336, TO_DATE('2020-05-0
insert into Employee (id, departId, profId, joinDate, salaryPerHour) values (213174246, 541, 5411327, TO_DATE('2021-11-2
insert into Employee (id, departId, profId, joinDate, salaryPerHour) values (213353976, 541, 5411328, TO_DATE('2021-09-1
insert into Employee (id, departId, profId, joinDate, salaryPerHour) values (213422783, 541, 5411329, TO_DATE('2019-05-1
insert into Employee (id, departId, profId, joinDate, salaryPerHour) values (213660538, 541, 54113210, TO_DATE('2019-04-
insert into Employee (id, departId, profId, joinDate, salaryPerHour) values (214531622, 541, 54113211, TO_DATE('2022-07-
insert into Employee (id, departId, profId, joinDate, salaryPerHour) values (215118975, 541, 54113212, TO_DATE('2020-12-
insert into Employee (id, departId, profId, joinDate, salaryPerHour) values (215277737, 541, 54113213, TO_DATE('2019-08-
insert into Employee (id, departId, profId, joinDate, salaryPerHour) values (215478697, 541, 54113214, TO_DATE('2023-04-
insert into Employee (id, departId, profId, joinDate, salaryPerHour) values (215558103, 541, 54113215, TO_DATE('2021-03-
insert into Employee (id, departId, profId, joinDate, salaryPerHour) values (216316883, 541, 54113216, TO_DATE('2021-04-
insert into Employee (id, departId, profId, joinDate, salaryPerHour) values (217162464, 541, 54113217, TO_DATE('2023-03-
insert into Employee (id, departId, profId, joinDate, salaryPerHour) values (217365718, 541, 54113218, TO_DATE('2023-05-
insert into Employee (id, departId, profId, joinDate, salaryPerHour) values (217733353, 541, 54113219, TO_DATE('2021-02-
insert into Employee (id, departId, profId, joinDate, salaryPerHour) values (219020575, 541, 54113220, TO_DATE('2019-09-
insert into Employee (id, departId, profId, joinDate, salaryPerHour) values (219211503, 541, 54113221, TO_DATE('2023-03-
insert into Employee (id, departId, profId, joinDate, salaryPerHour) values (219852723, 541, 54113222, TO_DATE('2019-10-
```

על אף זהה מתיואר כמו שDATE מຕואר – נצטרך לשנות את הערכים של התאריכים
ככה שאוקREL ימיר את זהה לערכים של DATE בבדיקה כהה –

```
TO_DATE('2021-07-23', 'YYYY-MM-DD')
TO_DATE('2023-01-07', 'YYYY-MM-DD')
```

כעת כל מה שנוטר לנו הוא להריץ את זהה.

ובום! יש לנו את זה הצלחנו ליצור 997 עובדים!
כמובן תתקפז שלא אומר כלום אבל לא אכפת לנו
כי נתקענו על זה כל כך הרבה זמן.
יששששששששש.

כעת נעברו לשלבים הבאים שכולים את המשמרות (לא משמרות המהיפה למרות שלפי המצב רוח של העובדים דזוקא נראה לי שהז' קרה אוטוטו) ואת השילוב של העובדים עם המשמרות הקיימות.

```
select count(*) from employee;
```

COUNT(*)
997

"רציתי לבוא לעבודה" אמרתי לבוס "אבל הכלב שלי אכל את הרבה קו" – טבלת המשמרות והרישום למשמרות (WorkShift, Set_On_Shift):

חיכים לא קלים לפעם, פתאום הבוס חוטף גננה וմבקש ממי לבוא לעבודה ב-3 לפנות בוקר.

למה מה קרה? מה אני עבד? אז מה אם נרשמתי למשמרת לילה צריך להתחשב באנשים שמנמנים במהלך העבודה.

שגעונות כאלה אולי תשמעו במיקרוסופט אבל במלון "זה לה ווי" היוקרט שלנו העובדים שלנו מרצוים מעובדתם – אין פלא כי הראשון שפתח את פיו על הדבר ישר

הוצא להורג 😊

אבל כעת אנו נטרוך לבבנות לעובדים החוצים שלנו משמרות שבהם יוכל לכתת רגליים ולשבור את הגב.

از הבא נתחיל we shall? –

ניצור טבלה פשוטה המכילה את זמני תחילת המשמרת וסופה.
שימוש לב – לא ניתן לגמור משמרת
לפני שנתחיל אותה 😊

```
CREATE TABLE WorkShift (
    start_time      TIMESTAMP NOT NULL,
    end_time        TIMESTAMP NOT NULL,
    CHECK (start_time < end_time),
    CONSTRAINT pk_WorkShift PRIMARY KEY (start_time)
)/
```

על מנת ליצור את המידע שנכניס אל הטבלה בסיס הנתונים שלנו אנחנו נעזריםשוב בח'ית המحمد שלנו פ'ית –

```

import datetime

# Initialize start time
start_time = datetime.datetime(2024, 4, 1, 7, 30)

# Prepare SQL insert statements
sql_inserts = []

for i in range(100000):
    # Calculate end time
    end_time = start_time + datetime.timedelta(hours=8)

    # Generate SQL insert statement
    sql_insert = f"INSERT INTO WorkShift (start_time, end_time) VALUES (TO_TIMESTAMP('{start_time.strftime('%Y-%m-%d %H:'})', TO_TIMESTAMP('{end_time.strftime('%Y-%m-%d %H:')}'))

    # Append SQL insert statement to list
    sql_inserts.append(sql_insert)

    # Increment start time by 8 hours for the next row
    start_time = end_time

with open ('insertWorkShiftInfo.sql', "w") as file:
    # Print SQL insert statements
    for sql in sql_inserts:
        file.write(sql + '\n')
|

```

ביקשנו (יפה אחרית הוא נושא) שיצור לנו רשימה של משמרות – 5000 במספרם - עם זמן התחלת וזמן סוף בתקופת הזמן מהחודש הרביעי של 2024 ועד הרחק הרחוק אל העתיד.

כל זמן מורכב מתאריך מסוים ושעת תחילת המשמרת – 7:30 או 15:30.
אורך כל משמרת הינו סך הכל 8 שעות בלבד.

```

kShift (start_time, end_time) VALUES (TO_TIMESTAMP('2024-04-01 07:30:00', 'YYYY-MM-DD HH24:MI:SS'), TO_TIMESTAMP('2024-
kShift (start_time, end_time) VALUES (TO_TIMESTAMP('2024-04-01 15:30:00', 'YYYY-MM-DD HH24:MI:SS'), TO_TIMESTAMP('2024-
kShift (start_time, end_time) VALUES (TO_TIMESTAMP('2024-04-01 23:30:00', 'YYYY-MM-DD HH24:MI:SS'), TO_TIMESTAMP('2024-
kShift (start_time, end_time) VALUES (TO_TIMESTAMP('2024-04-02 07:30:00', 'YYYY-MM-DD HH24:MI:SS'), TO_TIMESTAMP('2024-
kShift (start_time, end_time) VALUES (TO_TIMESTAMP('2024-04-02 15:30:00', 'YYYY-MM-DD HH24:MI:SS'), TO_TIMESTAMP('2024-
kShift (start_time, end_time) VALUES (TO_TIMESTAMP('2024-04-02 23:30:00', 'YYYY-MM-DD HH24:MI:SS'), TO_TIMESTAMP('2024-
kShift (start_time, end_time) VALUES (TO_TIMESTAMP('2024-04-03 07:30:00', 'YYYY-MM-DD HH24:MI:SS'), TO_TIMESTAMP('2024-
kShift (start_time, end_time) VALUES (TO_TIMESTAMP('2024-04-03 15:30:00', 'YYYY-MM-DD HH24:MI:SS'), TO_TIMESTAMP('2024-
kShift (start_time, end_time) VALUES (TO_TIMESTAMP('2024-04-03 23:30:00', 'YYYY-MM-DD HH24:MI:SS'), TO_TIMESTAMP('2024-
kShift (start_time, end_time) VALUES (TO_TIMESTAMP('2024-04-04 07:30:00', 'YYYY-MM-DD HH24:MI:SS'), TO_TIMESTAMP('2024-
kShift (start_time, end_time) VALUES (TO_TIMESTAMP('2024-04-04 15:30:00', 'YYYY-MM-DD HH24:MI:SS'), TO_TIMESTAMP('2024-
kShift (start_time, end_time) VALUES (TO_TIMESTAMP('2024-04-04 23:30:00', 'YYYY-MM-DD HH24:MI:SS'), TO_TIMESTAMP('2024-
kShift (start_time, end_time) VALUES (TO_TIMESTAMP('2024-04-05 07:30:00', 'YYYY-MM-DD HH24:MI:SS'), TO_TIMESTAMP('2024-
kShift (start_time, end_time) VALUES (TO_TIMESTAMP('2024-04-05 15:30:00', 'YYYY-MM-DD HH24:MI:SS'), TO_TIMESTAMP('2024-
kShift (start_time, end_time) VALUES (TO_TIMESTAMP('2024-04-05 23:30:00', 'YYYY-MM-DD HH24:MI:SS'), TO_TIMESTAMP('2024-
kShift (start_time, end_time) VALUES (TO_TIMESTAMP('2024-04-06 07:30:00', 'YYYY-MM-DD HH24:MI:SS'), TO_TIMESTAMP('2024-
kShift (start_time, end_time) VALUES (TO_TIMESTAMP('2024-04-06 15:30:00', 'YYYY-MM-DD HH24:MI:SS'), TO_TIMESTAMP('2024-
kShift (start_time, end_time) VALUES (TO_TIMESTAMP('2024-04-06 23:30:00', 'YYYY-MM-DD HH24:MI:SS'), TO_TIMESTAMP('2024-
kShift (start_time, end_time) VALUES (TO_TIMESTAMP('2024-04-07 07:30:00', 'YYYY-MM-DD HH24:MI:SS'), TO_TIMESTAMP('2024-
kShift (start_time, end_time) VALUES (TO_TIMESTAMP('2024-04-07 15:30:00', 'YYYY-MM-DD HH24:MI:SS'), TO_TIMESTAMP('2024-
kShift (start_time, end_time) VALUES (TO_TIMESTAMP('2024-04-07 23:30:00', 'YYYY-MM-DD HH24:MI:SS'), TO_TIMESTAMP('2024-
kShift (start_time, end_time) VALUES (TO_TIMESTAMP('2024-04-08 07:30:00', 'YYYY-MM-DD HH24:MI:SS'), TO_TIMESTAMP('2024-
kShift (start_time, end_time) VALUES (TO_TIMESTAMP('2024-04-08 15:30:00', 'YYYY-MM-DD HH24:MI:SS'), TO_TIMESTAMP('2024-
kShift (start_time, end_time) VALUES (TO_TIMESTAMP('2024-04-08 23:30:00', 'YYYY-MM-DD HH24:MI:SS'), TO_TIMESTAMP('2024-

```

והנה הערכים המג'ונרטים נרץ אוטם ב-PLSQL כמו שעשינו עד כה ... -

הנה לנו 5000 משמרות עבור עובדיםינו יקרים.

כǐ אַתְּ יֹדֵעַ מָה אָמַרְתָּ – "יִוְרֶר עֲבוֹדָה פָּחוֹת זָמָן לְחַשּׁוּב עַל הַעֲתִיד אוֹ מָקוֹם עֲבוֹדָה יוֹתֵר טוֹבָה".

נ.ב. אולי זה לא יעבוד לכם لكن כדאי לפחות את זה ל-5 קבצים של 1000 נתונים כל אחת.

```
SQL Output Statistics
select count(*) from WorkShift;
COUNT(*)
1 5000
```

cut נshedך את העובדים הקרים שלנו עם המשמרות היקרות שלהם.

על מנת לעשות את ניצור קובץ עם כל זמני תחילת המשמרות באמצעות הקוד הזה –

```
import datetime

start_time = datetime.datetime(2024, 4, 1, 7, 30)

sql_inserts = []

for i in range(5000):
    end_time = start_time + datetime.timedelta(hours=8)

    sql_insert = f"TO_TIMESTAMP('{start_time.strftime('%Y-%m-%d %H:%M:%S')}','YYYY-MM-DD HH24:MI:SS'),"
    sql_inserts.append(sql_insert)

    start_time = end_time

with open ('listOfStartHours.txt', "w") as file:
    # Print SQL insert statements
    for sql in sql_inserts:
        file.write(sql + '\n')
```

וככל מספרי הזרחות של העובדים – מה שכבר קיימים לנו – ונשמרו אותם בתיקייה על מנת להשתמש במלול הנתונים של PLSQL –

Name	Type	Size	Data
ID	NUMBER	9	PersonsId.EmployeeId
START_TIME	TIMESTAMP(6)	11	Starthours

וכמו קודם נריץ את המחולל זהה –

ובום יש לנו את זה אנשיים!

[18:25:00] 5000 records generated in 7.016 seconds

מנהלים הגדרה: אנשיים כמו שמרוייחים יותר מכך מסיבה פוליטית כדיות או אחרת – טבלת המנהלים והמגיסטים (Recruter, Manager):

כעת ניצור קבוצה חדשה – היא לא בדיק קבוצה ייחודית אבל למחשבה לעתיד היא יכולה להועיל אם נרצה להפריד בין מנהלים לשאר האנשים.

נחפש את כל המנהלים שקיים לנו בכל מחלקה ואת מספר המחלקה שלהם –

כעת ניצא את הערכים האלה אל קובץ CSV חיצוני שנשתמש בו על מנת להכניס את הערכים אל טבלת המנהלים

נעביר את הקובץ זהה אל Text importer שבתוכנה ונכנס אל טבלת המנהלים –

The screenshot shows the Oracle SQL Developer Data Pump Import interface. It displays the mapping between columns in the CSV file and the MANAGER table. The 'General' tab is selected, showing the owner (SYS), table name (MANAGER), and various import options like Overwrite duplicates, Delete records, and Truncate table. The 'Fields' tab shows the mapping: Field1 ID -> ID and Field2 DEPARTID -> MANAGEDDEPART. The 'Result Preview' tab shows a small sample of the data being imported.

ניתן לראות כי שם מנהל לא התפסה במהלך הכנסת הנתונים אל טבלת המנהלים.

The screenshot shows the results of a SQL query in the SQL tab of Oracle SQL Developer. The query selects the ID and department ID for employees whose profession title is either 'Manager' or 'Semi-Manager'. The results are displayed in a grid:

ID	DEPARTID
1	544
2	544
3	545
4	545
5	546
6	546
7	540
8	540
9	541
10	541
11	542
12	542
13	543
14	543

The screenshot shows the final SQL query in the SQL tab of Oracle SQL Developer. The query uses a subquery to find managers who have been assigned to departments they do not manage themselves. The subquery selects managers who have IDs different from their managed department IDs. The main query then joins this result with the MANAGER table to get the full list of managers and their department IDs.

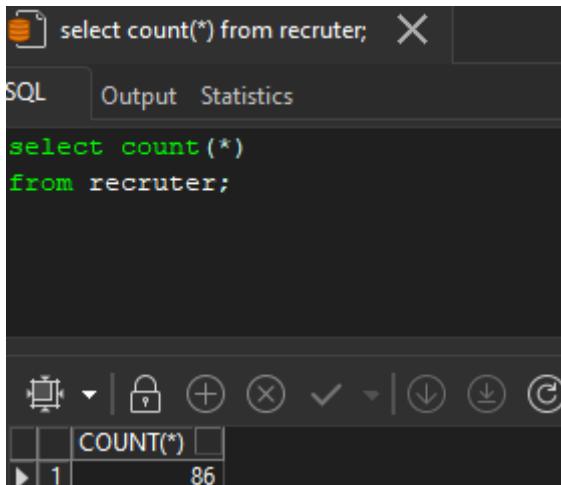
```

select t.id, t.departid
from (select id, departId
      from Employee natural join Profession
     where profTitle = 'Manager' or profTitle = 'Semi-Manager') t
  where (t.id,t.departId) !=
  (
  select *
    from manager m
   where m.id != id and m.manageddepartId != departId
  );
  
```

נעsha את אותו הדבר עבור המגיסטים – נמצא את רשימת כל המגיסטים וונשמר את מספר הזהות שלהם –

נינח את התוצאות וויצא לקובץ csv ונשתמש ב Text Importer על מנת להכניס את הנתונים אליו

וכעת לאחר כל זאת אנחנו רואים כי –



```
select count(*) from recruter;
```

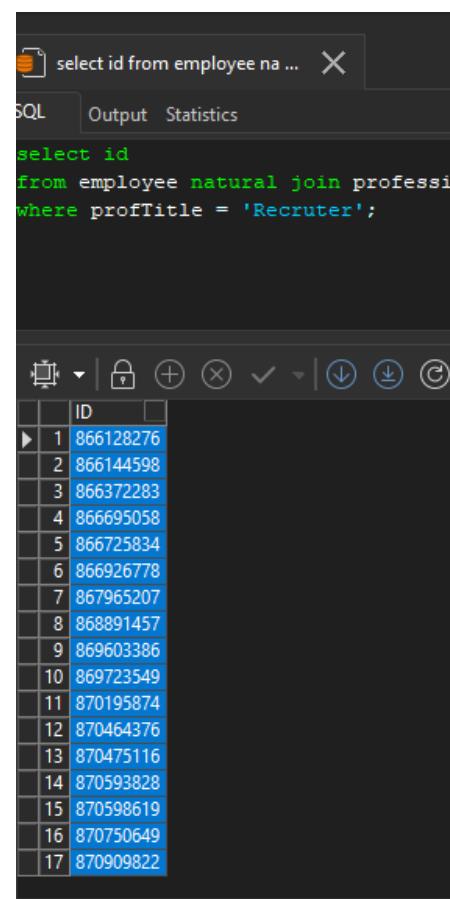
The screenshot shows a SQL query window with the following content:

```
SQL Output Statistics
select count(*)
from recruter;
```

Below the query results table:

	COUNT(*)
1	86

הוספנו את המגייסים היקרים שלנו אל טבלת המגייסים.
כעת כל מה שנותר לנו לעשות הוא ליצור את טבלת המועמדים היקרים שלנו וליצור את הקשר אל העבודה בה
הם מעוניינים להשתעבד עומד להיות כיף.



```
select id
from employee natural join professi
where profTitle = 'Recruter';
```

The screenshot shows a SQL query window with the following content:

```
SQL Output Statistics
select id
from employee natural join professi
where profTitle = 'Recruter';
```

Below the query results table:

ID
1 866128276
2 866144598
3 866372283
4 866695058
5 866725834
6 866926778
7 867965207
8 868891457
9 869603386
10 869723549
11 870195874
12 870464376
13 870475116
14 870593828
15 870598619
16 870750649
17 870909822

אם הייתה חזרה אחריה בזמן והייתי אומר למשהו ב1373** שאנשים רצים להשטעבד
בשביל אחרים הם היו סוקלים אותו על עצם היהי חיזיר – טבלת העובדים
העתידים וטבלת הקשר רוצה לעבוד ב... (FutureEmployee, Willing_To_Work_At):**

כמייטב המסתורת נתחילה דבר ראשון מטהlixir יצרת הטבלאות –

טבלת העובדים העתידיים מכילה בתוכה את
מספר זהות של המועמד.

```
CREATE TABLE FutureEmployee (
    fEmpId          NUMBER(9) NOT NULL,
    CONSTRAINT pk_FutureEmployee PRIMARY KEY (fEmpId),
    CONSTRAINT fk_FutureEmployee2 FOREIGN KEY (fEmpId)
        REFERENCES Person (Id),
    CONSTRAINT fk_FutureEmployee FOREIGN KEY (recId)
        REFERENCES Recruter (Id)
    ON DELETE CASCADE)
    /
```

```
CREATE TABLE Willing_To_Work_At (
    Id              NUMBER(9) NOT NULL,
    profId         NUMBER(9) NOT NULL,
    departId       NUMBER(3) NOT NULL,
    recId          NUMBER(9),
    dateOfPositionPlacement DATE NOT NULL,
    CONSTRAINT pk_Willing_To_Work_At PRIMARY KEY (Id,profId,departId),
    CONSTRAINT fk_Willing_To_Work_At FOREIGN KEY (Id)
        REFERENCES FutureEmployee (fEmpId)
    ON DELETE CASCADE,
    CONSTRAINT fk_Willing_To_Work_At2 FOREIGN KEY (profId,departId)
        REFERENCES Profession (profId,departId),
    CONSTRAINT fk_FutureEmployee FOREIGN KEY (recId)
        REFERENCES Recruter (Id)
    ON DELETE CASCADE)
    /
```

כפי שניתן לראות היא מכילה בתוכה את המספר זהות של המועמד (פז) את מספרי הזיהוי של העבודה שכלל את מספר התפקיד ומספר המחלקה שבה יעבד, את תאריך הגשת המועמדות ובנוסף היהת ואנחנו מדינה דמוקרטיות ולכן נתנו את האופציה לשועבדים עתידיים יגישו מועמדות לעבודות בלבד ללא עזרת המגייסים – לכן הכרחנו את התוכנה של המגייס להיות בעל ערך NULL.

cutת לערכיהם – כפי שזכרנו לכם הפרדנו בין העובדים למועמדים ובין המועמדים על ידי מגיסים וכאליה שלא.

cutת זה יבוא לנו ממש בשימוש טוב – נכנס את כל המועמדים אל FutureEmployee אך בהכנסת הערכים לקשר נפריד ביניהם כך יהיו לנו מערכת גם עובדים שלא הגיעו מועמדות על ידי המגייסים.

טבלת הקשר של העובדים המועמדים לatableת העובדות.

כפי שניתן לראות קשר זה הוא בין היחיד' שnitן להם טבלה משליהם היהת והוא קשר של רבים לרבים لكن לא ניתן פשוט לשמר את המפתח זר של הטבלה השנייה באחת מהטבלאות כי יכול להיות יותר מעבודה אחרת לכל מועמד ויותר ממועמד אחד לכל עבודה.

נתחל עם הכנסת המועמדים –
נשתמש בקובץ Future Employee שיש לנו בתיקית Text Employee ונתממש ב Importer

– Willing_To_Work_At

דבר ראשון נרצה לוודא איזה עבודות פנויות בחברה – הרি לא נרצה לקחת את העבודה למשהו אחר נכון? שואל רציני אי אפשר לדעת כבר בעולם הזה מה אתי ומה לא נrix את הפקודה הבאה כדי לקבל טבלת תוצאה עם העבודות הפנויות –

נו אтем יודעים כבר מה בא עכשו נכון? ניקח את כל הערכים מטבלת התוצאה, ניצא לCSV ואז נשמר את המאגרים באתר mockaroo ושם ניצור את הערכים עבור הטבלה זו.

אז קדימה בואו נעשה את זה כמו שאנו יודעים נלחץ על הכפתור בצד שמאל למעלה ונבחר את כל הטבלה ניצור קובץ CSV ונשמר.

ניצור קובץ נוסף שמכיל בתוכו רשימת מספרי זהות של כל המגייסים –

נ裏 את הפקודה הבאה שתראה לנו את רשימת כל המגייסים שנמצאים בקובץ הישיות "מגייס".

ניקח את הערכים מטבלת התוצאה זו וניצא לקובץ CSV ונשמר.

רשימת המועמדים - הן בררי מגייסים והן הבודדים לאו המגייסים - כבר נמצאים לנו בתיקיה יחד עם שאר קבצי הCSV لكن כל מה שנותר לנו הוא ליצור את המידע מהמאגרים הקיימים לנו.

ID
1 211317837
2 866128276
3 866144598
4 866372283
5 866695058
6 866725834
7 866926778
8 867965207
9 868891457
10 869603386
11 869723549
12 870195874
13 870464376

עליה לאתר mockaroo את כל המאגרים הנתונים לנו ונתחל ליצור את המידע עבור המועמדים בר' המגייסים –

Field Name	Type	Options
id	Dataset Column	Future Employees With Recruiters sequential blank: 0 % Σ X
profId	Dataset Column	Professions profId random blank: 0 % Σ X
departId	Dataset Column	Professions departId random blank: 0 % Σ X
dateOfPositionPlacement	Datetime	04/16/2023 to 03/15/2025 format: m/d/yyyy blank: 0 % Σ X
recId	Dataset Column	RecruitersId random blank: 0 % Σ X

Rows: 1000 Format: SQL Table Name: Willing_To_Work_At include CREATE TABLE

ນבוקש ליצור שורות מידע בהתאם למשרות הפתוחות בשימוש עם המגייסים החביבים שלנו בואו נראה איך זה יראה –

```

insert into Willing_To_Work_At (id, profId, departId, dateOfPositionPlacement, recId) values (799982564, '55022142', '550', '1/2/2024', '870464376');
insert into Willing_To_Work_At (id, profId, departId, dateOfPositionPlacement, recId) values (800675710, '54720169', '547', '2/27/2024', '888913939');
insert into Willing_To_Work_At (id, profId, departId, dateOfPositionPlacement, recId) values (801045087, '54310551', '543', '11/18/2024', '889005597');
insert into Willing_To_Work_At (id, profId, departId, dateOfPositionPlacement, recId) values (802147803, '54720189', '547', '5/29/2024', '866725834');
insert into Willing_To_Work_At (id, profId, departId, dateOfPositionPlacement, recId) values (802279299, '545106367', '545', '1/28/2025', '880722675');
insert into Willing_To_Work_At (id, profId, departId, dateOfPositionPlacement, recId) values (802329670, '541132381', '541', '5/13/2023', '880650316');
insert into Willing_To_Work_At (id, profId, departId, dateOfPositionPlacement, recId) values (802531550, '54316152', '543', '10/24/2024', '889005597');
insert into Willing_To_Work_At (id, profId, departId, dateOfPositionPlacement, recId) values (802568617, '547105226', '547', '8/10/2024', '888045871');
insert into Willing_To_Work_At (id, profId, departId, dateOfPositionPlacement, recId) values (802578684, '547105222', '547', '6/11/2024', '873603166');
insert into Willing_To_Work_At (id, profId, departId, dateOfPositionPlacement, recId) values (802785050, '54510190', '545', '8/12/2024', '881865530');
insert into Willing_To_Work_At (id, profId, departId, dateOfPositionPlacement, recId) values (802849520, '54810221', '548', '2/6/2025', '870750649');
insert into Willing_To_Work_At (id, profId, departId, dateOfPositionPlacement, recId) values (803122354, '547202109', '547', '11/14/2023', '871329079');
insert into Willing_To_Work_At (id, profId, departId, dateOfPositionPlacement, recId) values (803383466, '541132821', '541', '4/22/2024', '877724278');

```

נבצע שינוי קטן – כמו תמיד נוסף את הטקסט שאומר לאורקל להמיר לערך של DATE

```
Id, dateOfPositionPlacement) values (799982564, '54821172', '548', 876011722, TO_DATE('2023-01-20','YYYY-MM-DD'));
Id, dateOfPositionPlacement) values (800675710, '54417134', '544', 874679872, TO_DATE('2023-04-29','YYYY-MM-DD'));
Id, dateOfPositionPlacement) values (801045087, '545106258', '545', 879513631, TO_DATE('2023-03-30','YYYY-MM-DD'));
Id, dateOfPositionPlacement) values (802147803, '546193149', '546', 869723549, TO_DATE('2023-03-22','YYYY-MM-DD'));
Id, dateOfPositionPlacement) values (802279299, '54821293', '548', 870195874, TO_DATE('2023-09-25','YYYY-MM-DD'));
Id, dateOfPositionPlacement) values (802329670, '54210418', '542', 875860022, TO_DATE('2024-07-17','YYYY-MM-DD'));
Id, dateOfPositionPlacement) values (802531550, '54518133', '545', 870475116, TO_DATE('2023-12-05','YYYY-MM-DD'));
Id, dateOfPositionPlacement) values (802568617, '545181211', '545', 875812389, TO_DATE('2023-07-29','YYYY-MM-DD'));
Id, dateOfPositionPlacement) values (802578684, '54316299', '543', 879225385, TO_DATE('2024-06-10','YYYY-MM-DD'));
Id, dateOfPositionPlacement) values (802785050, '54821153', '548', 870909822, TO_DATE('2024-05-23','YYYY-MM-DD'));
Id, dateOfPositionPlacement) values (802849230, '54113215', '541', 881823143, TO_DATE('2023-08-07','YYYY-MM-DD'));
Id, dateOfPositionPlacement) values (803122354, '54720153', '547', 869723549, TO_DATE('2023-08-25','YYYY-MM-DD'));
Id, dateOfPositionPlacement) values (803383466, '54518196', '545', 889005597, TO_DATE('2023-08-27','YYYY-MM-DD'));
Id, dateOfPositionPlacement) values (803423324, '545106388', '545', 881823143, TO_DATE('2024-06-22','YYYY-MM-DD'));
```

משהו זהה וכעת נריץ ונכנים את המידע אליו תוך טבלת הקשר.

וכעת נעשה את כמעט אותו הדבר עם המועמדים שאינם ברוי מגייסים –

Field Name	Type	Options
id	Dataset Column	Future Employees Without Recruiters sequential blank: 0 % Σ X
profId	Dataset Column	Professions profId random blank: 0 % Σ X
departId	Dataset Column	Professions departId random blank: 0 % Σ X
dateOfPositionPlace	Datetime	01/08/2023 to 07/26/2024 format: yyyy-mm-dd blank: 0 % Σ X

השינוי היחיד הוא שלא הוספנו שדה של מספר זהות המגייס, כמו כן גם שינויו את המאגר מידע וכעת אנחנו משתמשים במאגר מידע של המועמדים ללא מגייסים.

נוריד את הסקריפט ונשנה אותו שוב עם תוספת קטנה של TO_DATE על מנת להמיר לערך DATE

```
values (799982564, '54821172', '548', 876011722, TO_DATE('2023-01-20','YYYY-MM-DD'));
values (800675710, '54417134', '544', 874679872, TO_DATE('2023-04-29','YYYY-MM-DD'));
values (801045087, '545106258', '545', 879513631, TO_DATE('2023-03-30','YYYY-MM-DD'));
values (802147803, '546193149', '546', 869723549, TO_DATE('2023-03-22','YYYY-MM-DD'));
values (802279299, '54821293', '548', 870195874, TO_DATE('2023-09-25','YYYY-MM-DD'));
values (802329670, '54210418', '542', 875860022, TO_DATE('2024-07-17','YYYY-MM-DD'));
values (802531550, '54518133', '545', 870475116, TO_DATE('2023-12-05','YYYY-MM-DD'));
values (802568617, '545181211', '545', 875812389, TO_DATE('2023-07-29','YYYY-MM-DD'));
```

נრיץ את הקובץ וכעת יש לנו את העובדים המועמדים, כאלה שהגיסו מועמדות וכolumbia שלא, כolumbia עם מגייסים וכolumbia שלא – החיים הם לא שchor לben, בכל שchor יש אישיה בordon קטן שמסתתר בו.

והנה הordon קטן שלנו – חמוץ ומספק סוף לסוגת יצירת הטבלאות ואכלוסיהם 😊😊.

```
select count(*) from willing_to_work_at;
+-----+
| COUNT(*) |
+-----+
|      2000 |
+-----+
```

