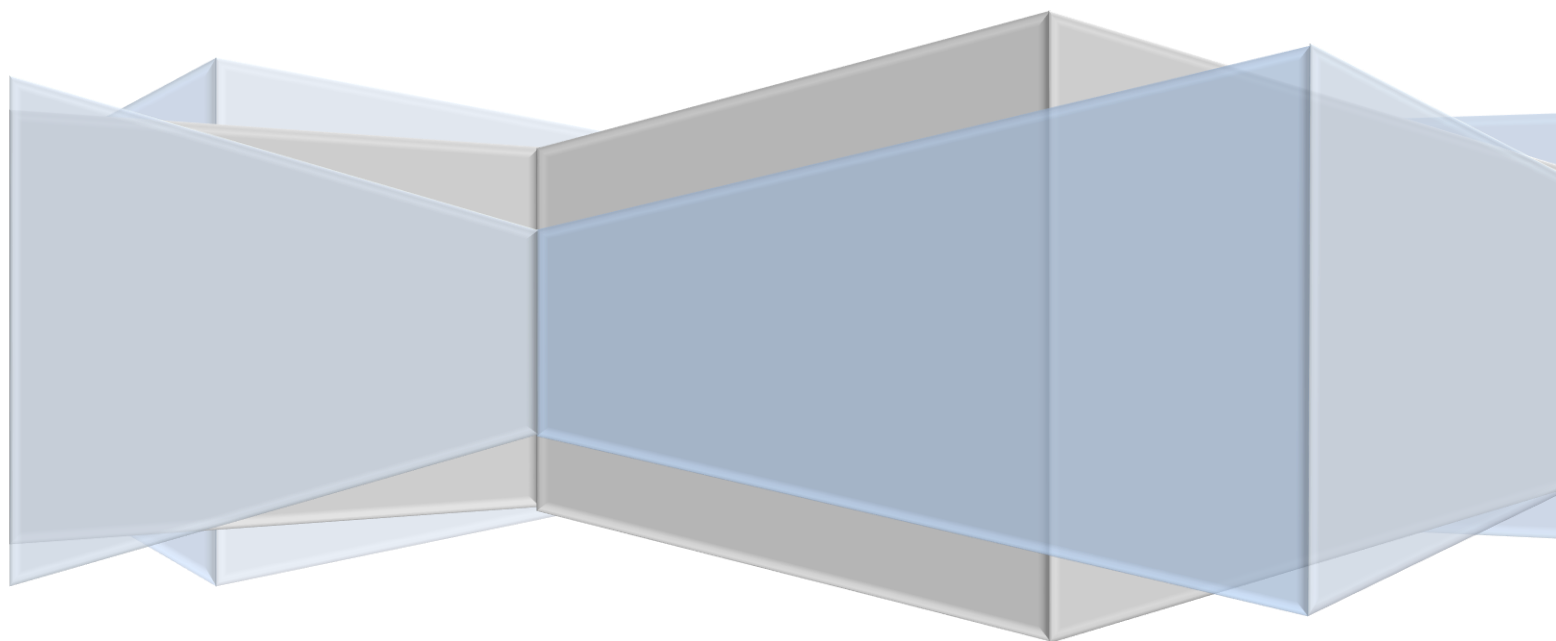


בס"ד

# מיני פרויקט בבסיסי נתונים

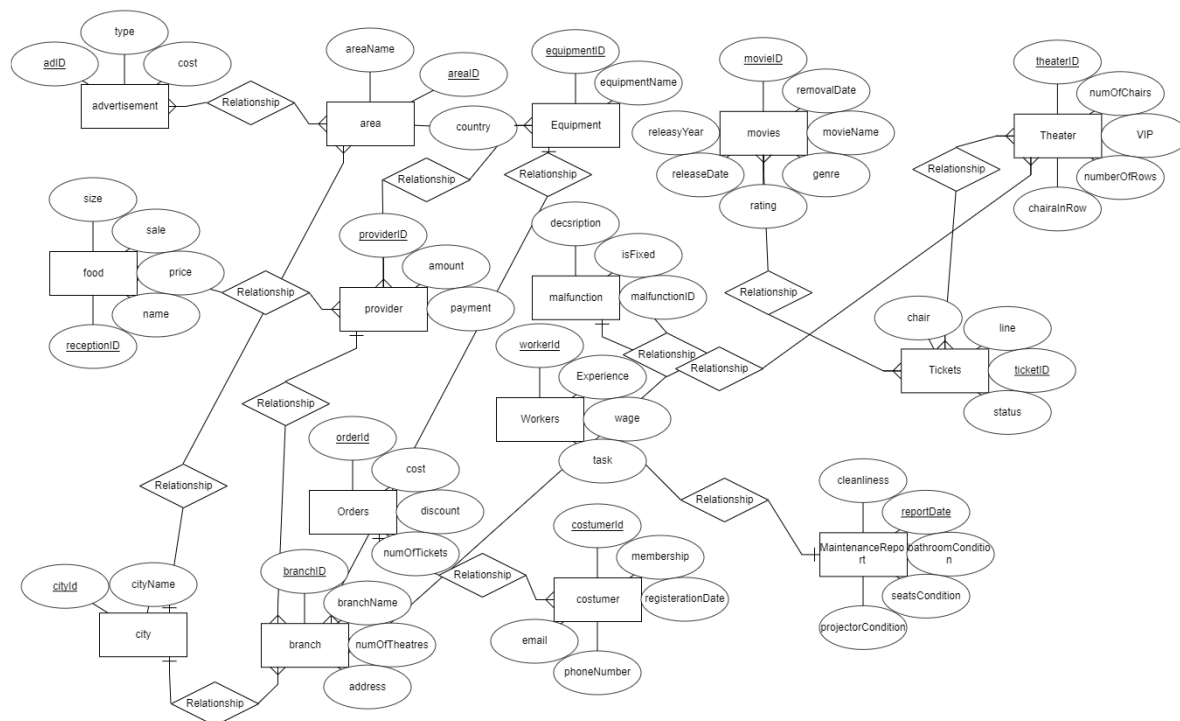
הוד לוי 211516562

אליאב ברוך 302379375



# עבודת הכנה והכרת התוכנה

## תרשים ERD



## תיאור הישויות והקשרים

### ישויות

- Branch - מאופיין ב- מזהה של הסניף, שם, כתובת, מספר אולמות, מזהה של העיר, מזהה של הציוד ומזהה של הספק.
- Theatre - מאופיין ב- מזהה של האולם, מספר כסאות באולם, VIP או לא, מספר שורות, מספר כסאות בשורה.
- Area - מאופיין ב- מספר אזור ובשם האזור ושם המדינה.
- City מאופיין ב- מספר העיר ובשם העיר ומספר אזור.

### קשרים

- לכל סניף- יש ספק יחיד, יכול להיות לו מספר מסוים של תקלות, ומספר מסוים של ציוד ומספר מסוים של אולמות ועיר יחידה.
- לכל אולם- יכול להיות מספר מסוים של כרטיסים ומספר מסוים של תקלות.
- לכל עיר- יכולים להיות הרבה אולמות והיא יכולה להיות באזור יחיד.
- לכל אזור – יכולות להיות כמה ערים וכמה פרסומות.

### נרמול הטבלאות

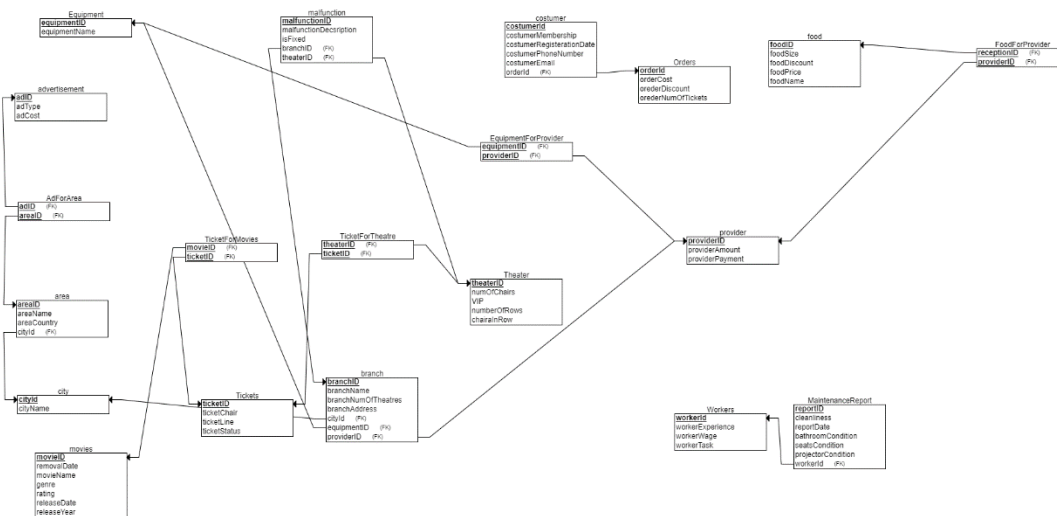
- Branch (branchID, branchName, branchNumOfTheatres, BranchAddress, cityID, equipmentID, providerID)
- Theatre (theatreID, numOfChairs, VIP, numberOfRows, chairInARow)
- CityName (cityID, cityName, areaID)
- Area (areaID, areaName, areaCountry)

### פרוקים

היחסים עומדים ב- 3NF וב- BCNF : מכיוון שבכל טבלה, התלויות הפונקציונאליות הלא-טריוויאליות הן מהמפתח אל תכונות נוספות לכן מתקיים שלכל  $X \rightarrow Y$ ,  $X$  הוא מפתח ולכן הם עומד בתנאים.

## תרשים DSD

כאן ניתן לראות את ה-DSD שהפקנו מתרשים ה-ERD שיצרנו.

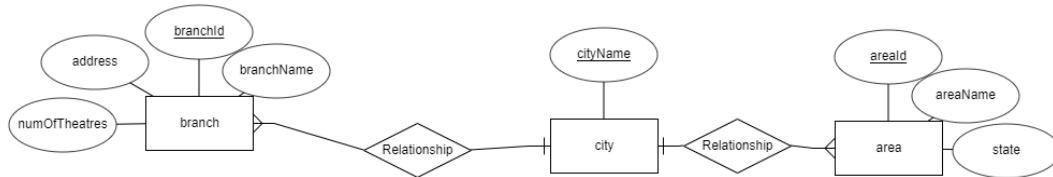


# הפרויקט שלנו

אנחנו התמקדנו ביחסים בין אלומות, סניפים, ערים ואיזורים.

## תרשים ERD

כפי שהזכרנו, במחלקה שלנו, ישנן 3 ישויות: מתקנים, חשבונות ובקשות אספקה. בשלב הראשון יצרנו תרשים ERD שיתאר את הקשרים בין הישויות הללו ואת התכונות שלהן.



## תיאור הישויות והקשרים

### ישויות

- Branch – ישות זאת אחראית על כל המתקנים הנמצאים במערכת.  
ישות זאת הינה חזקה, כיוון שיכולה להתקיים ללא תלות בישות אחרת.
  - Branch\_ID – מספר מזהה של הסניף (PK)
  - Branch\_name – שם הסניף
  - Address – כתובת הסניף
  - Num\_Of\_Theatres – מספר אולמות בסניף
  - City\_ID – מספר מזהה של העיר (FK)
- City – ישות זאת אחראית על כל בקשות האספקה הנמצאים במערכת.
- ישות זאת הינה חזקה, כיוון שיכולה להתקיים ללא תלות בישות אחרת. (נעשו שינויים ממה שמוצג למעלה)
  - City\_id – מספר מזהה של העיר (PK)
  - City\_Name – שם העיר
  - Area\_id – מספר מזהה של האיזור (FK)
- Area – ישות זאת אחראית על כל החשבונות הנמצאים במערכת.
- ישות זאת הינה חזקה, כיוון שיכולה להתקיים ללא תלות בישות אחרת.
  - Area\_id – מספר מזהה של האיזור (PK)

- Area\_name – שם האיזור
- Country – מדינה

### קשרים

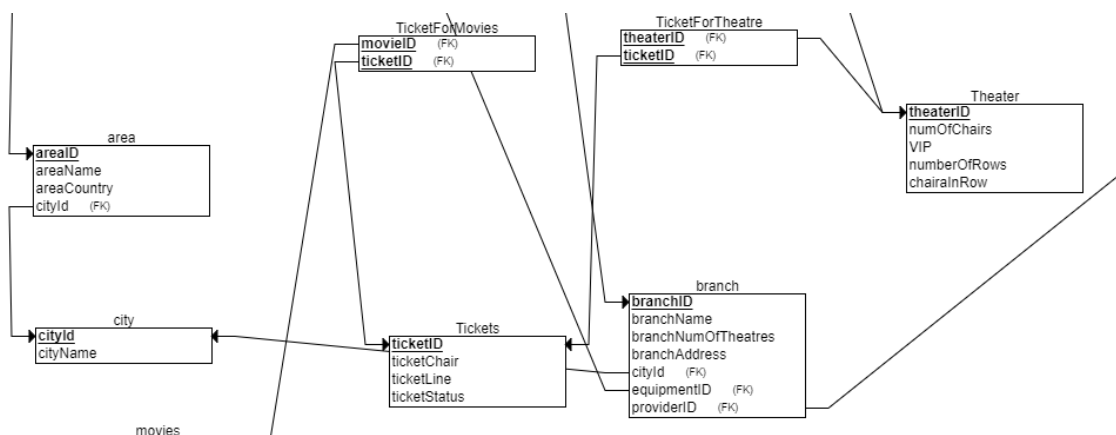
- BranchCity – הקשר בין branch לבין city. הקשר הוא M:1 משום שיכול להיות לעיר אחת הרבה סניפים, אבל סניף אחד שייך לעיר אחת.
- CityArea – הקשר בין city לבין area. הקשר הוא M:1 משום שיכול להיות לאיזור אחד הרבה ערים, אבל עיר אחת שייכת לאיזור אחד בלבד.

### נרמול הטבלאות

היחסים עומדים ב- 3NF וב- BCNF : מכיוון שבכל טבלה, התלויות הפונקציונאליות הלא-טריוויאליות הן מהמפתח אל תכונות נוספות לכן מתקיים שלכל  $X \rightarrow Y$ ,  $X$  הוא מפתח ולכן הם עומד בתנאים.

## תרשים DSD

על פי תרשים ה- ERD ועל ידי הבנת הקשרים בין הישויות, יצרנו תרשים DSD עבור החלק שלנו במערכת: סניפים ואולמות.



## יצירת הטבלאות

אחרי שהבנו כיצד בסיס הנתונים צריך להראות בצורה מדויקת, מה תכיל כל טבלה ומהם הקשרים בין כל הטבלאות, ניגשנו ליצירת הטבלאות בפועל בעזרת פקודות create table.

יצרנו קוד לייצור הטבלאות באמצעות export SQL של האתר erdPlus, יצרנו קובץ SQL ואז העתקנו את קוד ה- SQL של כל טבלה אל תוכנת ה-plsql לשם יצירת הטבלאות בפועל. כל הקודים נמצאים בGIT של אורי ארליך.

## הכנסת נתונים

השתמשנו בData generator של ה-PLSQL. קבצים בGIT של אורי ארליך.

## שאלות SQL

לאחר שיצרנו את בסיס הנתונים והכנסנו לתוכו מידע, כתבנו כמה שאלות מעניינות על מנת לתשאל אותו.

- 1. בדקנו איזה סניפים יש בישראל

```
select branchID
from oehrlich.branch natural join oehrlich.city natural join oehrlich.area
where areaCountry = 'Israel'
```

	BRANCHID
1	396
2	486
3	604
4	26
5	71
6	273
7	1462
8	1506
9	1522
10	1566
11	1622
12	1699
13	1060
14	1242
15	1917
16	2158
17	2173
18	2397
19	2558
20	2694
21	2234
22	2264
23	1771
24	1776
25	1792
26	1830
27	1896
28	3485
29	3611
30	3694
31	2914
32	2777

- 2. בדקנו כמה סניפים יש בבאר שבע

```
select count (*)
from oehrlich.branch natural join oehrlich.city
where cityName = 'Beer Sheva'
```

	COUNT(*)
1	0



### 3. בדקנו באילו איזורים יש סניפים שיש בהם פחות מ-20 תקלות

```
select areaName
from oehrlich.city natural join oehrlich.area natural join oehrlich.branch b
where 20 >= (select count(branchid)
             from oehrlich.malfunction m
             where b.branchid = m.branchid)
```

	AREANAME
1	Trevino
2	Haysbert
3	Hutch
4	Hector
5	Nelson
6	Oldman
7	Johnson
8	Haynes
9	Bruce
10	Folds
11	Davidson
12	Johansen
13	Belle
14	Uggams
15	Holeman
16	Griffiths
17	Eckhart
18	Ruiz
19	Chandler
20	Del Toro
21	Copeland
22	Gracie
23	Hauser
24	Vai
25	Rio
26	Colman
27	Giannini
28	Koteas
29	Apple

### 4. בדקנו באילו ערים יש סניפים שיש בהם תקלות שלא טופלו

```
select distinct cityName
from oehrlich.city natural join oehrlich.branch natural join oehrlich.malfunction m
where m.isfixed = 'no'
```

	CITYNAME
1	Chely
2	Steven
3	Frankie
4	Cherry
5	Morris
6	Bernie
7	Mena
8	Pablo
9	Joely
10	Joe
11	Richard
12	Stanley
13	Michelle
14	Larenz
15	Aaron
16	Terry
17	Laurence
18	Celia
19	Ray
20	Reese
21	Daryl
22	Courtney
23	Carl
24	Arturo
25	Rose
26	Holly
27	Edwin
28	Lydia
29	Avril
30	Jeffery
31	Denis
32	Nancy

5. בדקנו באילו אולמות שהם ויאייפי ומספר השורות בהם הוא מקסימלי יש לפחות כרטיס אחד שהוא בסטטוס "פעיל"

```

select theaterid
from (select theaterid, numberofrows
      from oehrlich.theater
      where vip = 'YES' and numberofrows >= ALL(select t.numberofrows from oehrlich.theater t)) natural join oehrlich.ticketfortheatre natural join oehrlich.ticketstatus
where ticketstatus = 'Active'

```

	THEATERID
1	16889
2	11027
3	1877
4	7220
5	3293
6	1712
7	15746
8	2013
9	17046
10	16435
11	14773
12	7796
13	2475
14	8762
15	3219
16	8304
17	3505
18	6239
19	6298
20	18156
21	17954
22	14816
23	2673
24	9205
25	14670
26	6096
27	8194
28	15538
29	12188

6. בדקנו אילו סרטים מוקרנים באולמות שבהם יש יותר כסאות מתקלות

```

select distinct moviename
from oehrlich.movies natural join oehrlich.ticketformovies natural join oehrlich.tickets natural join oehrlich.theater t
where numofchairs > (select count(m.malfunctionid)
                    from oehrlich.malfunction m
                    where m.isfixed = 'no' and m.theaterid = t.theaterid)

```

	MOVIE NAME
1	USA Environmental Management
2	MedAmicus
3	Lydian Trust
4	Pinnacle Staffing
5	HealthScribe
6	Flake-Wilkerson Market Insights
7	InfoVision Consultants
8	Intel Corp.
9	IntegraMed America
10	Catamount Constructors
11	U.S. Energy Services
12	SPS Commerce
13	Cherokee Information Services
14	Universal Solutions
15	Cura Group
16	CCF Holding
17	American Vanguard
18	Nature's Cure
19	NoBrainerBlinds.com
20	Virbac
21	Kingston
22	Solution Builders
23	AOL Time Warner
24	Strategic Management Initiatives
25	Dankoff Solar Products
26	Zaiq Technologies
27	Genex Technologies
28	Great Lakes Media Technology
29	Consultants' Choice
30	Custom Solutions International
31	Manhattan Associates
32	Advanced Neuromodulation
33	FirstFed America Bancorp
34	Reckitt Benckiser

7. בדקנו איזה כרטיס הזמין מקום באולם מעבר למספר השורות הקיימות באותו אולם

```
select ticketid, theaterid
from oehrlich.theater natural join oehrlich.tickets natural join oehrlich.ticketfortheatre
where ticketline > numberofrows
```

	TICKETID	THEATERID
1	699	16845
2	702	6507
3	706	14028
4	707	7804
5	714	18377
6	718	554
7	727	5967
8	730	8322
9	735	4950
10	742	16327
11	744	18542
12	744	13175
13	748	15600
14	753	16589
15	757	4424
16	758	11473
17	761	17834
18	763	13541
19	767	12101
20	768	11057
21	770	13231
22	777	6094
23	779	6773
24	784	17468
25	784	774
26	786	13918
27	797	1511
28	799	10572
29	812	9296
30	819	4155
31	820	15577
32	830	8217
33	833	5401

8. בדקנו איזה סרטים שבהם אדם הזמין כרטיס לכסא בקצה האולם יצאו משנת 1994 ואילך

```
select moviename
from (select moviename, releaseyear
from oehrlich.theater natural join oehrlich.ticketfortheatre natural join oehrlich.tickets natural join oehrlich.movies natural join oehrlich.ticketformovies
where ticketchair = chairainrow and ticketline = numberofrows) m
where m.releaseyear >= 1994
```

	MOVIE NAME
1	Mission West Properties ...
2	Typhoon ...

## אינדקסים

אינדקסים עוזרים למצוא במהירות גדולה יותר נתונים שנשמרו בטבלאות בבסיס הנתונים. אפשר לדמות את האינדקסים כמו מראה מקום בספר. במקום שנקרא את כל הספר כדי למצוא את מה שאנחנו מחפשים נלך למראה מקום שיראה לנו את כל המקומות שבהם מוזכר הנושא שאנחנו מחפשים. השימוש באינדקסים יחסוך לנו זמן ויהפוך את תהליך החיפוש ליעיל יותר. מהבחינה הזו האינדקסים בטבלאות של ה-SQL זהים לאינדקס בספר.

במידה ולא נגדיר אינדקס לטבלה אז בכל שאילתה על הטבלה השאילתה תגרום למעבר על כל הרשומות בטבלה עד שתמצא את כל הרשומות העונות למה שחיפשנו. כשנגדיר אינדקס מתאים אז החיפוש יהיה מהיר יותר כי הפניה לבסיס הנתונים תגרום לזה שמנוע החיפוש בבסיס הנתונים יפנה קודם לאינדקס וילך לרשומות המתאימות על פי מה שרשום באינדקס.

לכן יצרנו אינדקסים שמקצרים את תהליך ביצוע השאילתות.

את כל הבדיקות עשינו ללא cache ע"י שסגרנו את התוכנה ( ולפעמים כיבינו את המחשב ), על מנת שזה לא יושפע מה-cache.

האינדקסים הנ"ל:

```
create index IDX_MALFUNCTION_IS_FIXED on oehrlich.malfunction(isfixed);

create index IDX_BRANCH_BRANCHID on oehrlich.branch(branchid);

create index IDX_AREA_AREACOUNTRY on oehrlich.area(areacountry);
```

1. שאילתה מספר 1: שאלנו שאילתה המחזירה טבלה של שמות המספרים המזהים של כל הסניפים שנמצאים בישראל בעקבות בקשה של הציבור בישראל לדעת אילו סניפים קולנוע של הרשת המצליחה שלנו נמצאים בארץ הקודש.

ישנם 49,275 סניפים, ומתוכם נמצאים בישראל 391. שיפרנו את זמן ההרצה של השאילתה בערך פי 3!

האינדקס שיצרנו היה על המדינה "areaCountry" ולכן כשחיפשנו בשאילתה את המספרים המזהים של הסניפים שנמצאים בישראל היה מהיר יותר לקבל את התוצאה כי ההסתברות לקבל שם של מדינה היא גבוהה יותר.

לפני אינדוקס	אחרי אינדוקס
0.237	0.086

2. שאילתה מספר 3: שאלנו שאילתה המחזירה טבלה של כל שמות האיזורים שיש בהם לפחות סניף אחד שיש בו מעל 20 תקלות, שהרי ידוע שסניף עם מעל ל-20 תקלות מקבל ציון נכשל במדד הבטיחות של מכון התקנים של האו"ם ונידון לסגירה ולקנס גבוה מאוד. ישנם 6,808 איזורים, ומתוכם 30 עונים על השאילתה. שיפרנו את זמן הריצה בערך פי 2!

האינדקס שיצרנו היה על המספר המזהה של הסניפים, לכן מבחינה הסתברותית יש לנו פחות רשומות שחוזרות על עצמן מאשר מזהה שהוא רציף ושונה מרשומה אחת לשנייה ולכן זמן החיפוש היה מהיר יותר.

לפני אינדוקס	אחרי אינדוקס
0.197	0.096

3. שאילתה מספר 6: שאלנו שאילתה המחזירה טבלה של כל שמות הסרטים שמוקרנים באולמות שבהם יש יותר כיסאות מתקלות, כי כך אם נניח שכל התקלות הן בכיסאות נוכל לשער שלא בכל הכיסאות יש תקלות וכך נוכל לאשר את הקרנת הסרט.

ישנם 7,195 סרטים, ומתוכם 697 עונים על השאילה. שיפרנו את זמן הריצה בזמן זעיר מאוד שלא ניתן לראות השפעה אמיתית.

האינדקס שיצרנו היה על העמודה is\_fixed ולכן מבחינה הסתברותית יש לנו פחות רשומות זהות מאשר עמודות אחרות בטבלה ולכן זמן החיפוש היה מעט קצר יותר.

לפני אינדוקס	אחרי אינדוקס
34.372	34.223

## Views

VIEWS הם טבלאות וירטואליות. VIEWS מכילים הגדרות של עמודות וסוגי מידע שאותן עמודות יכולות להכיל. ההבדל בין הטבלאות לבין ה-VIEWS הוא שבטבלאות נשמרים נתונים באופן פיזי ואילו ב-VIEWS הנתונים לא נשמרים באופן פיזי בתוכם אלא הם רק מציגים נתונים הנשמרים בטבלאות. לכן לא ניתן לעדכן או להוסיף נתונים ל-VIEWS כפי שעושים לטבלאות.

(1) יצרנו view עם עמודות theatroid, ticketid, branchid כיוון שרבות מהשאלות שלנו מתייחסות לעמודות אלו ולכן יעיל יותר לעבוד עם שלושת עמודות אלו באופן נפרד, בלי התייחסות לטבלאות המלאות.

```
CREATE or replace view Theatre_Branch_Ticket as
select theaterID, branchID, ticketID
from oehrlich.malfunction natural join oehrlich.ticketfortheatre
```

דוגמה לשימוש בVIEW בשאלתה מספר 11

(2) יצרנו view עם עמודות theatroid, ticketid, movieid כיוון שרבות מהשאלות שלנו מתייחסות לעמודות אלו ולכן יעיל יותר לעבוד עם שלושת עמודות אלו באופן נפרד, בלי התייחסות לטבלאות המלאות.

```
CREATE or replace view Movie_Ticket_Theatre as
select MovieName, TheaterID, TicketID
from oehrlich.TicketForMovies natural join Theatre_Branch_Ticket
natural join oehrlich.movies
```

(3) יצרנו view עם עמודות areaName, areald, branchid כיוון שחלק מהשאלות שלנו מתייחסות לעמודות אלו ולכן יעיל יותר לעבוד עם שלושת עמודות אלו באופן נפרד, בלי התייחסות לטבלאות המלאות.

```
CREATE or replace view Area_Branch as
select areaName, areaId, branchId
from oehrlich.area natural join oehrlich.city natural join
oehrlich.branch
```

(4) יצרנו view עם עמודות VIP, theatroid כיוון שחלק מהשאלות שלנו מתייחסות לעמודות אלו ולכן יעיל יותר לעבוד עם שתי עמודות אלו באופן נפרד, בלי התייחסות לטבלאות המלאות.

```
CREATE or replace view Ticket_VIP as
select ticketID, VIP
from oehrlich.theater natural join oehrlich.ticketfortheatre
```

## פונקציות

פונקציה בשפת SQL היא צורה מיוחדת של פקודה אשר מבצעת פעולות שונות על הנתונים בבסיס הנתונים ומחזירה ערך.

פונקציה מספר 1:

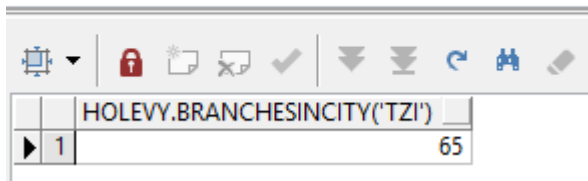
בהינתן שם של עיר, הפונקציה מחזירה מספר סניפי קולנוע שיש באותה עיר

```
create or replace function BranchesInCity(city_name in string) return
number is
    FunctionResult number;
begin
    select count(branchID) into FunctionResult
    from oehrlich.city c natural join oehrlich.branch
    where c.cityname = city_name
    group by c.cityname;

    return(FunctionResult);
end BranchesInCity;
```

דוגמת הרצה:

```
select holevy.branchesincity('Tzi')
from dual;
```



	HOLEVY.BRANCHESINCITY('TZI')
1	65

פונקציה מספר 2:

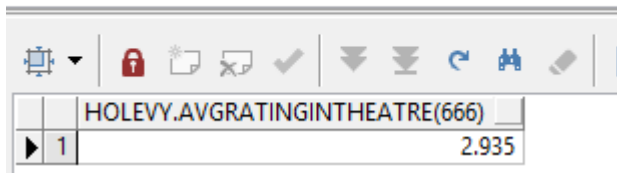
בהינתן מספר מזהה של אולם, הפונקציה מחזירה את ממוצע הרייטינג של הסרטים שהוקרנו באותו אולם

```
create or replace function AvgRatingInTheatre(theatre_id in number)
return float is
    FunctionResult float;
begin
    select AVG(rating) into FunctionResult
    from oehrlich.movies natural join oehrlich.ticketformovies natural
join oehrlich.tickets natural join oehrlich.ticketfortheatre natural
join oehrlich.theater t
    where theaterid = theatre_id
    group by theaterid;

    return(FunctionResult);
end AvgRatingInTheatre;
```

דוגמת הרצה:

```
select holevy.avgratingintheatre(666)
from dual;
```



HOLEVY.AVGRATINGINTHEATRE(666)	
1	2.935

## פרוצדורות

פרוצדורה בשפת SQL היא צורה מיוחדת של פקודה אשר מבצעת פעולות שונות על הנתונים בבסיס הנתונים.

פרוצדורה מספר 1:

הפרוצדורה מעדכנת כל כרטיס שמספר השורה שלו גדול יותר ממספר השורות המקסימלי למספר השורות המקסימלי

```
create or replace procedure UpdateWrongRow is
cursor t_tickets is
select *
from tickets
for update;

rec_tickets tickets%rowtype;
update_line tickets.ticketline%type;

begin
  open t_tickets;
  loop
    fetch t_tickets into rec_tickets;
    exit when t_tickets%notfound;
    update_line := 0;
    if(rec_tickets.ticketline > 25)then
      update tickets t
      set t.ticketline = 25
      where current of t_tickets;
    end if;
  end loop;
  close t_tickets;
end UpdateWrongRow;
```



תוצאת ההרצה של שאילתה 7 (הבודקת מקרים של חריגה זו) לפני ביצוע הפרוצדורה:

```
select ticketid, theaterid
from oehrlich.theater natural join oehrlich.tickets natural join oehrlich.ticketfortheatre
where ticketline > numberofrows
```

	TICKETID	THEATERID
1	699	16845
2	702	6507
36	840	2/93

0:01 holevy@labdbwin 6880 rows selected in 1.170 seconds

תוצאת ההרצה של שאילתה 7 (הבודקת מקרים של חריגה זו) לאחר ביצוע הפרוצדורה:

```
select ticketid, theaterid
from oehrlich.theater natural join oehrlich.tickets natural join oehrlich.ticketfortheatre
where ticketline > numberofrows
```

	TICKETID	THEATERID
--	----------	-----------

## פרוצדורה מספר 2:

בהינתן שם של סרט, הפרוצדורה מכניסה למשתנה ticket\_count את מספר הכרטיסים שנמכרו לסרט הנתון.

```
create or replace procedure MovieData(movie_name in string,
ticket_count out number) is
begin
    select count(ticketID) into ticket_count
    from oehrlich.movies m natural join oehrlich.ticketformovies
    natural join oehrlich.tickets
    where m.movieName = movie_name;
end MovieData;
```

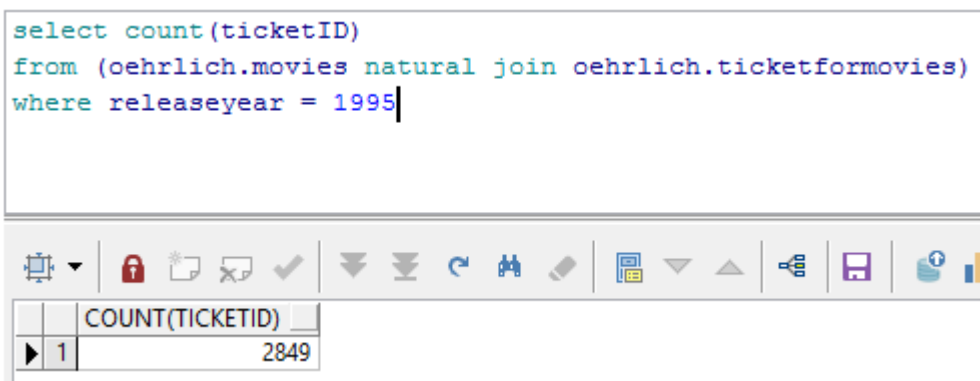
## שאלות SQL נוספות

קיבלנו תוספת של ישויות שאיתן עלינו לעבוד (סרטים, אולמות וכרטיסים), וכעת נעשה שאלות SQL נוספות שיכללו ישויות אלו.

שאלת מספר 9:

בדקנו כמה כרטיסים נמכרו בשנת 1995

```
select count(ticketID)
from (oehrlich.movies natural join oeherlich.ticketformovies)
where releaseyear = 1995|
```

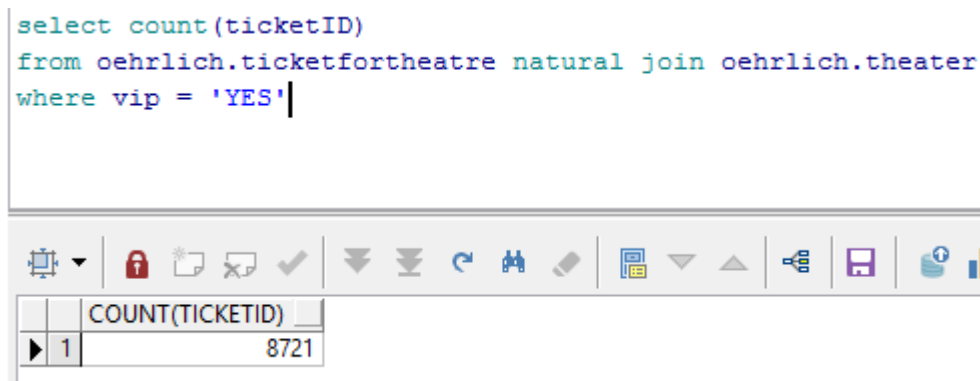


	COUNT(TICKETID)
1	2849

שאלת מספר 10:

בדקנו כמה כרטיסים נמכרו לאולם VIP

```
select count(ticketID)
from oeherlich.ticketfortheatre natural join oeherlich.theater
where vip = 'YES'|
```



	COUNT(TICKETID)
1	8721

שאלת מספר 11:

בדקנו כמה כרטיסים נמכרו בכל עיר

```
select cityName, count(ticketID)
from oehrlich.city natural join Theatre_Branch_Ticket natural join oehrlich.branch
group by cityName
```

	CITYNAME	COUNT(TICKETID)
1	Mena	32
2	Gene	21
3	Gin	15
4	Dom	11
5	Ellen	15
6	Freda	15

שימוש בVIEW שיצרנו (למעלה) לעומת אי שימוש בו (למטה)

```
select cityName, count(ticketID)
from oehrlich.city natural join oehrlich.branch natural join oehrlich.malfunction natural join oehrlich.theater natural join oehrlich.ticketfortheatre
group by cityName
```

	CITYNAME	COUNT(TICKETID)
1	Mena	32
2	Gene	21
3	Gin	15
4	Dom	11
5	Ellen	15
6	Freda	15
7	Jeffery	8
8	Morris	7