
מיני פרויקט בבסיסי נתונים

נושא: פנאי - קולנוע



מגישות:

וינר רחלי – 325265221, viderracheli@gmail.com
מייפן נעמה – 214429193, naamameypen@gmail.com

תוכן עניינים

3	תיאור מילולי של המערכת.....
3	תיאור מילולי של הטבלאות.....
4	הפונקציונליות העיקרית של המערכת
5	תרשים ERD.....
6	תרשימי DSD.....
7	הצגת Create tables ופקודות Desc.....
10	הכנסת נתונים - Data generator.....
11	הכנסת נתונים - Python Script.....
12	הכנסת נתונים - mockaroo.....
13	גיבוי ושיחזור

דוח הפרוייקט שלב א

פרויקט על ארגון בתחום הפנאי פנאי - קולנוע.

תיאור המערכת:

מערכת ניהול הקולנוע נועדה לטפל בהיבטים התפעוליים של בית קולנוע. זה מקל על ניהול סרטים, זמני הצגה, קולנוע, מנויים ומכירת כרטיסים. המערכת שמה לה למטרה לייעל את תפעול הקולנוע, לשפר את חווית הלקוח ולייעל את ניצול המשאבים. על ידי שמירה על רישומים מפורטים ומאפשרת תזמון יעיל, המערכת מבטיחה תפקוד חלק ומשפרת את היעילות הכוללת של פעולות הקולנוע.

נתונים - טבלאות:

1. **סרטים:** מזהה סרט, כותרת, ז'אנר, במאי, תאריך יציאה, משך, דירוג.

טבלה זו מאחסנת מידע על הסרטים המוצגים בקולנוע, כולל תעודת הזהות, הכותרת, הז'אנר, הבמאי, תאריך היציאה, משך הזמן והדירוג שלהם.

2. **תאריכים:** DateID, HourH, DayD, DateD.

טבלה זו עוקבת אחר התאריכים והשעות שבהם הסרטים מתוכננים להיות מוצגים.

3. **אולמות:** TheaterID, TheaterName, Capacity.

טבלה זו מכילה מידע על בתי הקולנוע בתוך הקולנוע, כולל תעודת הזהות, השם ויכולת הישיבה של כל אולם.

4. **לוחות זמנים:** ScheduleID, TheaterID, AvailableSeats, MovieID, DateID.

טבלה זו מקשרת בין סרטים, תאריכים ובתי קולנוע, ומציינת איזה סרט מוצג באיזה קולנוע באיזו שעה וכמה מושבים זמינים לכל הצגה.

5. **מנויים:** מזהה מנוי, אחוזי הנחה, תאריך תפוגה.

טבלה זו מאחסנת מידע על המנויים, כולל תעודת הזהות שלהם, אחוז ההנחה ותאריך תפוגה של המנוי.

6. **מכירות כרטיסים:** TicketID, Price, IsSold, ScheduleID, SubscriberID.

טבלה זו מתעדת את מכירת הכרטיסים, כולל מזהה כל כרטיס, מחיר, סטטוס מכירה והקשרים עם לוחות זמנים ומנויים ספציפיים.

פונקציות עיקריות של המערכת:

1. ניהול סרטים:

- הוספה, עדכון והסרה של סרטים.
- מעקב אחר פרטי סרט כגון ז'אנר, במאי, תאריך יציאה, משך ודירוג.

2. תזמון מועדי הופעה:

- הגדרת זמני הקרנה לסרטים.
- הקצאת בתי קולנוע לסרטים שונים.
- ניהול המושבים הפנויים לכל שעת הופעה.

3. ניהול אולמות:

- שמירה על מידע על כל אולם בתוך הקולנוע.
- ניהול קיבולת וזמינות האולם.

4. ניהול מנויים:

- ניהול מידע מנויים.
- החלת הנחות למנויים.
- מעקב אחר תאריכי תפוגה של מנוי.

5. מכירת כרטיסים:

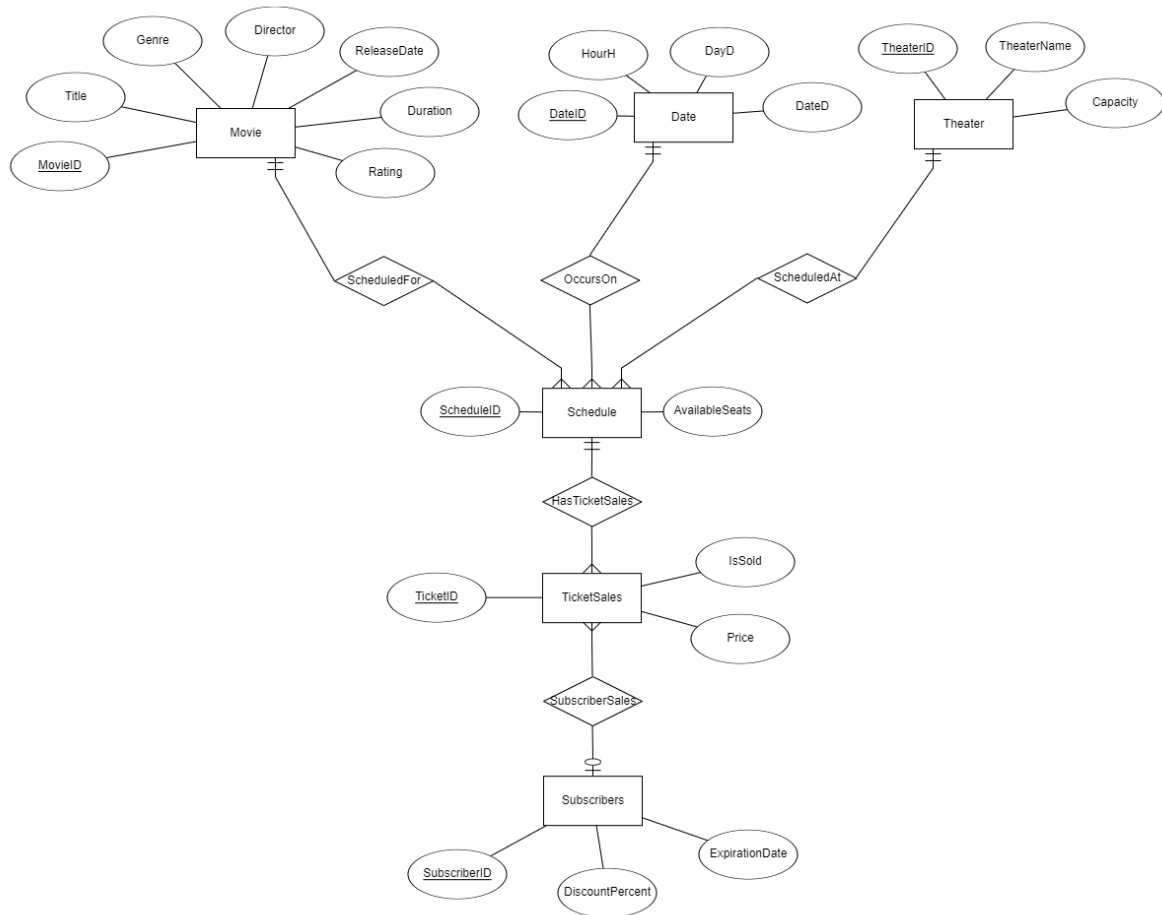
- מכירת כרטיסים למועדי הופעה קבועים.
- מעקב אחר מכירת כרטיסים וזמינות.
- החלת הנחות מנויים על מכירת כרטיסים.

6. ניהול תאריך וזמן:

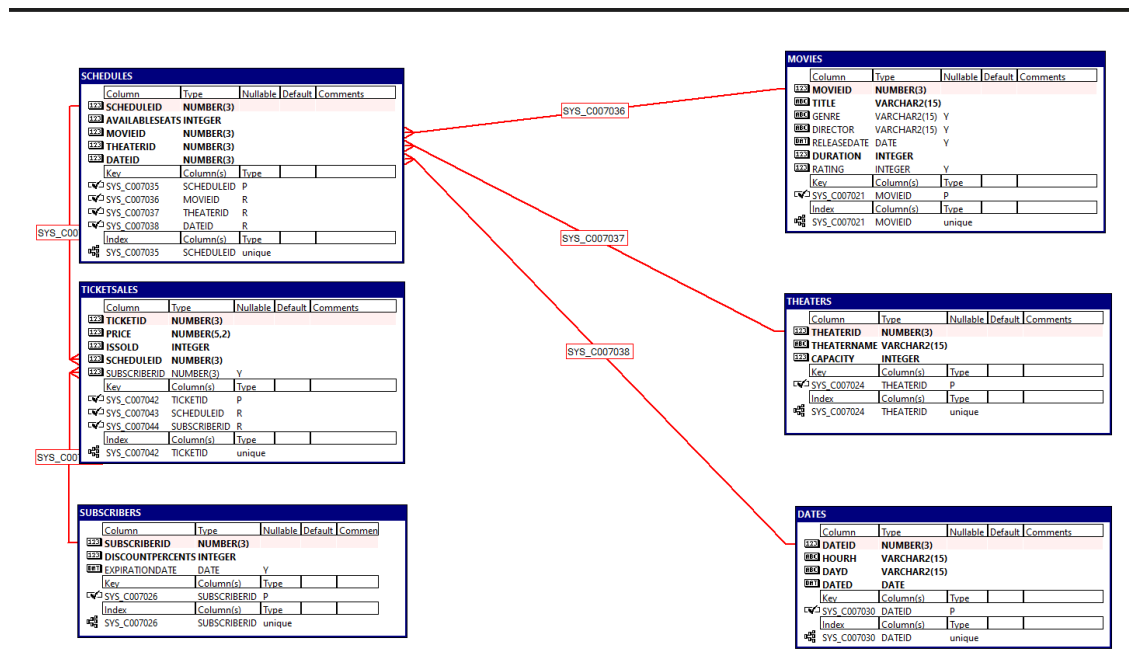
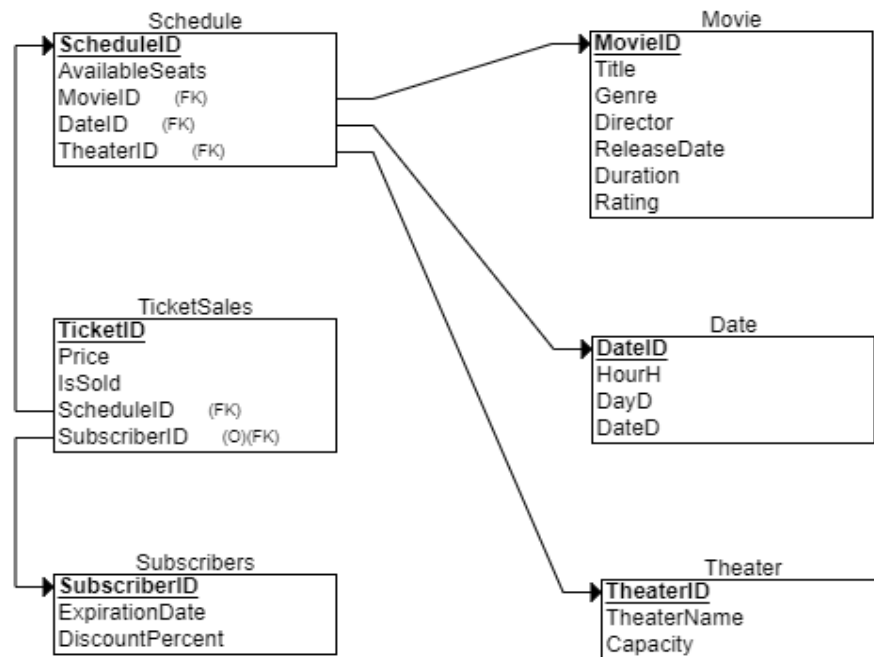
- רישום של כל התאריכים והשעות שבהם מוצגים סרטים.
- הבטחת תזמון מדויק והימנעות מעימותים.

תרשימי ה ERD ו ה DSD:

ERD:



DSD:



הצגת Create tables ופקודות Desc:

1. Movies:

```
CREATE TABLE Movies (  
    MovieID NUMERIC(3) PRIMARY KEY,  
    Title VARCHAR2(100) NOT NULL,  
    Genre VARCHAR2(100),  
    Director VARCHAR2(100),  
    ReleaseDate DATE,  
    Duration INT NOT NULL,  
    Rating INT  
);
```

```
SQL> desc Movies;  
Name          Type          Nullable Default Comments  
-----  
MOVIEID       NUMBER(3)  
TITLE         VARCHAR2(100)  
GENRE         VARCHAR2(100) Y  
DIRECTOR      VARCHAR2(100) Y  
RELEASEDATE   DATE          Y  
DURATION      INTEGER  
RATING        INTEGER      Y
```

2. Dates:

```
CREATE TABLE Dates (  
    DateID NUMERIC(3) PRIMARY KEY,  
    HourH VARCHAR2(100) NOT NULL,  
    DayD VARCHAR2(100) NOT NULL,  
    Dated DATE NOT NULL  
);
```

```
SQL> desc Schedules  
Name          Type          Nullable Default Comments  
-----  
SCHEDULEID    NUMBER(3)  
THEATERID     NUMBER(3)  
AVAILABLESEATS INTEGER  
MOVIEID       NUMBER(3)  
DATEID        NUMBER(3)
```

:Theaters .3

```
CREATE TABLE Theaters (  
    TheaterID NUMERIC(3) PRIMARY KEY,  
    TheaterName VARCHAR2(30) NOT NULL,  
    Capacity INT NOT NULL  
);
```

```
SQL> desc theaters  
Name          Type          Nullable Default Comments  
-----  
THEATERID     NUMBER(3)  
THEATERNAME   VARCHAR2(100)  
CAPACITY      INTEGER
```

:Schedules .4

```
CREATE TABLE Schedules (  
    ScheduleID NUMERIC(3) PRIMARY KEY,  
    AvailableSeats INT NOT NULL,  
    MovieID NUMERIC(3) NOT NULL,  
    TheaterID NUMERIC(3) NOT NULL,  
    DateID NUMERIC(3) NOT NULL,  
    FOREIGN KEY (MovieID) REFERENCES Movies(MovieID),  
    FOREIGN KEY (TheaterID) REFERENCES Theaters(TheaterID),  
    FOREIGN KEY (DateID) REFERENCES Dates(DateID)  
);
```

```
SQL> desc Schedules  
Name          Type          Nullable Default Comments  
-----  
SCHEDULEID    NUMBER(3)  
THEATERID     NUMBER(3)  
AVAILABLESEATS INTEGER  
MOVIEID       NUMBER(3)  
DATEID        NUMBER(3)
```

:Subscribers .5

```
CREATE TABLE Subscribers (  
    SubscriberID NUMERIC(3) PRIMARY KEY,  
    DiscountPercents INT NOT NULL,  
    ExpirationDate DATE  
);
```



```
SQL> desc Subscribers
Name                Type                Nullable Default Comments
-----
SUBSCRIBERID        NUMBER(3)
DISCOUNTPERCENTS   INTEGER
EXPIRATIONDATE       DATE                Y
```

:TicketSales .6

```
CREATE TABLE TicketSales (
  TicketID NUMERIC(3) PRIMARY KEY,
  Price NUMERIC(5, 2) NOT NULL,
  IsSold CHAR(1) DEFAULT 'N' CHECK (IsSold IN ('Y', 'N')) NOT NULL,
  ScheduleID NUMERIC(3) NOT NULL,
  SubscriberID NUMERIC(3),
  FOREIGN KEY (ScheduleID) REFERENCES Schedules(ScheduleID),
  FOREIGN KEY (SubscriberID) REFERENCES Subscribers(SubscriberID)
);
```

```
SQL> desc TicketSales
Name                Type                Nullable Default Comments
-----
TICKETID            NUMBER(3)
PRICE               NUMBER(5,2)
ISSOLD              CHAR(1)                'N'
SCHEDULEID          NUMBER(3)
SUBSCRIBERID        NUMBER(3)            Y
```

הכנסת נתונים לטבלאות ב3 שיטות:

1. Data generator: (TicketSales, Schedules)

TICKETSALES					
Owner	Table	Number of records			
RACHELI	TICKETSALES	400			
Name	Type	Size	Data	Master	
TICKETID	NUMBER	3	Sequence(100)	...	▼
PRICE	NUMBER	5,2	List(30, 30, 30, 30, 50, 90)	...	▼
ISSOLD	CHAR	1	List(N)	...	▼
SCHEDULEID	NUMBER	3	List(select scheduleid from schedules)	...	▼
SUBSCRIBERID	NUMBER	3	List(select subscriberid from subscribers)	...	▼
*				...	▼

SCHEDULES					
Owner	Table	Number of records			
RACHELI	SCHEDULES	400			
Name	Type	Size	Data	Master	
SCHEDULEID	NUMBER	3	Sequence(100)	...	▼
AVAILABLESEATS	NUMBER		List(0)	...	▼
MOVIEID	NUMBER	3	List(select MOVIEID from MOVIES)	...	▼
THEATERID	NUMBER	3	List(select THEATERID from THEATERS)	...	▼
DATEID	NUMBER	3	List(select DATEID from DATES)	...	▼
*				...	▼

```

1195 insert into RACHELI.SCHEDULES (SCHEDULEID, AVAILABLESEATS, MOVIEID, THEATERID, DATEID)
1196 values (498, 0, 424, 297, 190);
1197
1198 insert into RACHELI.SCHEDULES (SCHEDULEID, AVAILABLESEATS, MOVIEID, THEATERID, DATEID)
1199 values (499, 0, 120, 428, 228);
1200
1201 UPDATE Schedules s
1202 SET s.AvailableSeats = (
1203     SELECT t.Capacity
1204     FROM Theaters t
1205     WHERE t.TheaterID = s.TheaterID
1206 );
1207
1208 commit;
1209

```

לאחר הכנסת מקומות הישיבה כ 0 בכל הטבלאות, הוספנו בסוף הקובץ עדכון לערך תכולה, שיתאים לאולם שנבחר על פי ה TheaterID ניקח את התכולה של כל שורה בטבלה Theaters.

2. Python Script: (Theaters, Dates)

```
1 def read_flower_names(filename):
2     with open(filename, 'r') as file:
3         flowers = [line.strip() for line in file.readlines()]
4     return flowers
5
6 def generate_plsql_script(flowers, num_records, output_filename):
7     theater_id_start = 111
8     min_capacity = 20
9     max_capacity = 300
10
11     with open(output_filename, 'w') as file:
12         file.write("BEGIN\n")
13         for i in range(num_records):
14             theater_id = theater_id_start + i
15             theater_name = flowers[i % len(flowers)]
16             capacity = random.randint(min_capacity, max_capacity)
17             file.write(f"INSERT INTO THEATERS (THEATERID, THEATERNAME, CAPACITY) VALUES ({theater_id}, '{theater_name}', {capacity});\n")
18         file.write("END;\n")
19         file.write("/\n")
20
21 if __name__ == "__main__":
22     flowers = read_flower_names('flower_names.txt')
23     num_records = 400
24     output_filename = 'insert_theaters.sql'
25     generate_plsql_script(flowers, num_records, output_filename)
26     print(f"PL/SQL script '{output_filename}' generated successfully.")
```

```
1 from datetime import datetime, timedelta
2
3 # Function to generate sequential dates
4 def generate_sequential_dates(start_date, num_days):
5     return [start_date + timedelta(days=i) for i in range(num_days)]
6
7 # Define hours and days
8 hours = ["14:00", "17:00", "20:00", "23:00"]
9
10 # Start date for the records
11 start_date = datetime(2024, 6, 1)
12 num_days = 120 # Number of days to generate
13 total_records = 400 # Total number of unique records to generate
14
15 # Generate sequential dates
16 sequential_dates = generate_sequential_dates(start_date, num_days)
17
18 # Calculate how many records per day are needed
19 records_per_day = total_records // num_days
20
21 # Generate 400 unique records
22 records = []
23 ur = set()
24 date_id = 100 # Starting DateID
25
26 for day_date in sequential_dates:
27     day = day_date.strftime('%a').upper()
28     date_d = day_date.strftime('%d-%m-%Y')
29     hour_count = 0
30
```

```
11 for hour in hours:
12     if len(records) >= total_records:
13         break
14     # Use a tuple to ensure uniqueness
15     record = (date_id, hour, day, date_d)
16     unique_key = (hour, day, date_d)
17
18     # Add record to the set if the unique combination does not exist
19     if unique_key not in ur:
20         ur.add(unique_key)
21         records.append(record)
22         date_id += 1
23         hour_count += 1
24
25 # Write the insert statements to a SQL file
26 with open('date_insertion.sql', 'w') as file:
27     for record in records:
28         date_id, hour, day, date_d = record
29         insert_statement = f"INSERT INTO DATES (DATEID, HOURH, DAYD, DATED) VALUES ({date_id}, '{hour}', '{day}', TO_DATE('{date_d}', 'DD-MM-YYYY'));"
30         file.write(insert_statement + '\n')
```

3. mockaroo: (Subscribers, Movies)

Subscribers

Field Name	Type	Options	
SubscriberIDcketID	Digit Sequence	###	blank: 0 % Σ ×
DiscountPercents	Custom List	10,20,30,40,50	random blank: 0 % Σ ×
ExpirationDate	Datetime	05/20/2024 to 05/20/2030 format: dd-mm-yyyy blank: 0 % Σ ×	

+ ADD ANOTHER FIELD GENERATE FIELDS USING AI...

Rows: 400 Format: SQL Table Name: Subscribers ☐ Include CREATE TABLE

Append Dataset: choose a dataset...

Movies

Field Name	Type	Options	
MovieID	Digit Sequence	###	blank: 0 % Σ ×
Title	Movie Title	blank: 0 % Σ ×	
Genre	Movie Genres	blank: 0 % Σ ×	
Director	Full Name	blank: 0 % Σ ×	
ReleaseDate	Datetime	05/20/2000 to 05/20/2024 format: m/d/yyyy blank: 0 % Σ ×	
Duration	Number	min: 60 max: 180 decimals: 0 blank: 0 % Σ ×	
Rating	Number	min: 1 max: 10 decimals: 0 blank: 0 % Σ ×	

+ ADD ANOTHER FIELD GENERATE FIELDS USING AI...

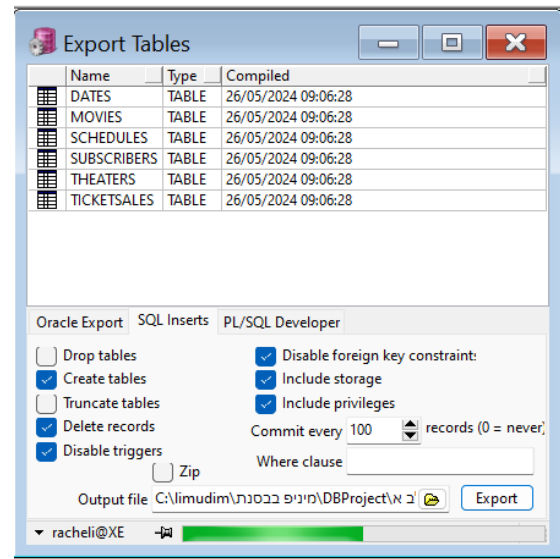
Rows: 100 Format: SQL Table Name: Movies ☐ Include CREATE TABLE

עבור שדה מטיפוס תאריך זה הקוד שכתבנו:

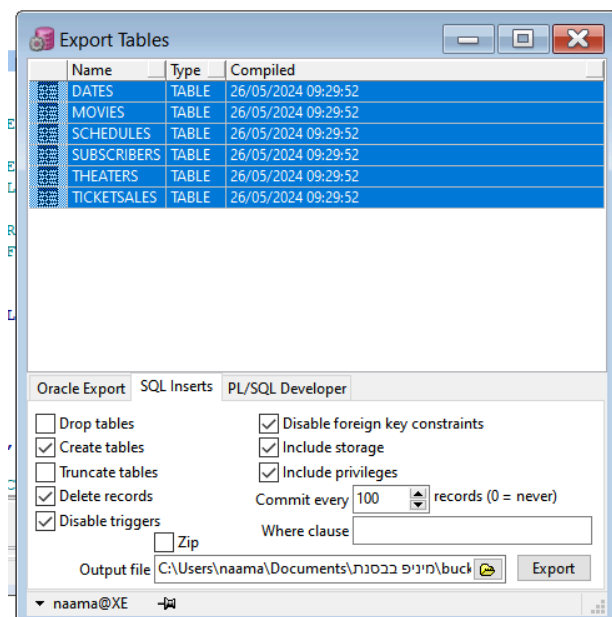
```
code(concat(code("to_date('",this,code("'", 'dd-mm-yyyy')"))
```

גיבוי ושחזור:

גיבוי:



שחזור:



```
Execute C:\limudim\מיוניפ בבסנת\DBProject\א\שלב\BackUp.sql
Dialog Editor
Connected to Oracle Database 11g Express Edition Release 11.2.0.2.0
Connected as rachel@XE

SQL>
PL/SQL Developer import file
Created on 2024 מאי 26 יום 11:26:26 by vider
Creating DATES...
Creating MOVIES...
Creating THEATERS...
Creating SCHEDULES...
Creating SUBSCRIBERS...
Creating TICKETSALES...
Disabling triggers for DATES...
Disabling triggers for MOVIES...
Disabling triggers for THEATERS...
Disabling triggers for SCHEDULES...
Disabling triggers for SUBSCRIBERS...
Disabling triggers for TICKETSALES...
Disabling foreign key constraints for SCHEDULES...
alter table SCHEDULES disable constraint SYS C007227
```

```
Execute C:\limudim\מיוניפ בבסנת\DBProject\א\שלב\BackUp.sql
Dialog Editor
100 records committed...
200 records committed...
300 records committed...
400 records loaded
Loading MOVIES...
100 records committed...
200 records committed...
300 records committed...
400 records committed...
401 records loaded
Loading THEATERS...
100 records committed...
200 records committed...
300 records committed...
400 records loaded
Loading SCHEDULES...
100 records committed...
200 records committed...
300 records committed...
400 records loaded
Loading SUBSCRIBERS...
100 records committed...
200 records committed...
300 records committed...
400 records committed...
401 records loaded
Loading TICKETSALES...
100 records committed...
200 records committed...
300 records committed...
400 records loaded
Disabling foreign key constraints for SCHEDULES
```