



# פרויקט בבסיסי נתונים- בית מלון

מגישות: לינוי ידעי ושירה גייר

## Contents

2.....	בית מלון
2.....	פירוט הישויות:
2.....	פירוט קשרים:
3.....	תרשימים:
3.....	תרשימים ERD:
3.....	תרשימים DSD
4.....	Create tables
5.....	DESC
6.....	הכנסת נתונים לטבלה בשלושה דרכים שונות:
6.....	1. data Generator
6.....	2. קובץ טקסט:
7.....	3. אתר אינטרנט
7.....	פעולת הגיבוי
8.....	פעולת שחזור הנתונים
9.....	שאלות
9.....	ארבעה שאלות:
11.....	ארבעה שאלות עם פרמטרים:
13.....	אילוצים
14.....	שאלות עידכון
16.....	שאלות מחיקה
18.....	גיבוי 2
19.....	תוכניות
19.....	תוכנית 1:
21.....	תוכנית 2:
24.....	גיבוי 3:
25.....	שלב 4: Error! Bookmark not defined.
25.....	DSD
25.....	פירוט מסקנות מה DSD ליצירת ה ERD:
26.....	ה ERD:
27.....	פקודות עיצוב:
28.....	ה DSD המשולב:

לינוי ידעי 213120942 linoy.y02@gmail.com

[shiragayer.12](mailto:shiragayer.12)

שירה גייר 214309700 [com](mailto:shiragayer.12)

## בית מלון

מקבל הזמנות של אורחים ע"י הזמנת חדרים, וסוגי פעילויות שכל לקוח הזמין ומחשב את תשלום ההזמנות הכולל.

בנוסף מציין העסקת עובדים ומקשר את עבודתם לכל פעילות שיש למלון להציע.

### פירוט הישויות:

לקוחות-אורחי המלון, לקוח מזוהה לפי ת"ז יש לו שם, מספר פלאפון, ואמייל.

חדרים-חדרי מלון שמזוהה לפי מספר חדר בהתאם לסוג החדר.

פעילויות-מספר של פעילויות שיש למלון, לכל פעילות יש ת"ז שם, מחיר ותיאור הפעילות.

עובדים-כל עובד מזוהה לפי ת"ז ולכל עובד יש תפקיד ושם.

הזמנה-לכל הזמנה יש ת"ז תאריך כניסה ותאריך עזיבה.

תשלום-מזהה תשלום, תאריך תשלום ומחיר ההזמנה.

### פירוט קשרים:

Resevation-לקוח מבצע הזמנת/ות אירוח למלון.

BookingRooms-כל הזמנה כוללת מספר חדרים וסוגם בהתאם לבקשת הלקוח.

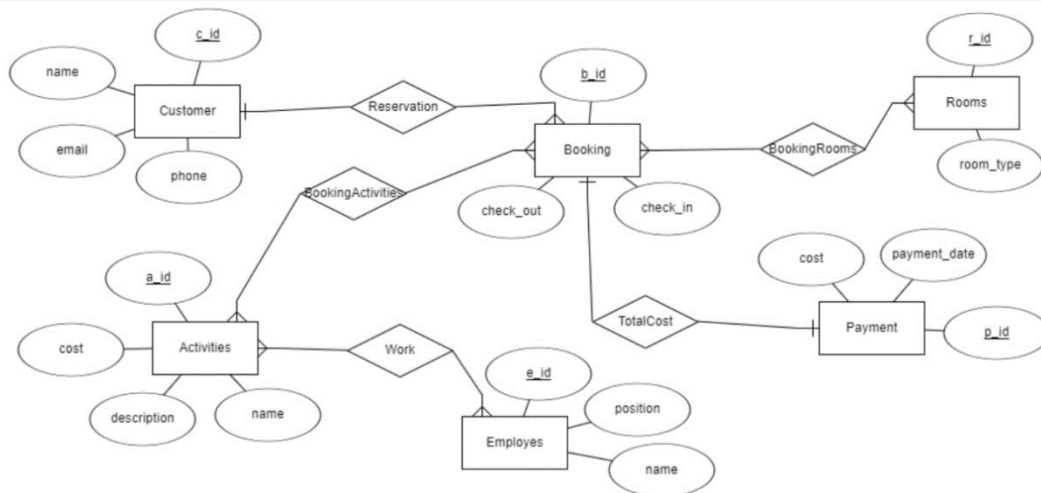
BookingActivitis-כל הזמנה כוללת מספר פעילויות וסוגן בהתאם להזמנת הלקוח.

Work-כל עובד אחראי על פעילות בהתאם לתפקידו.

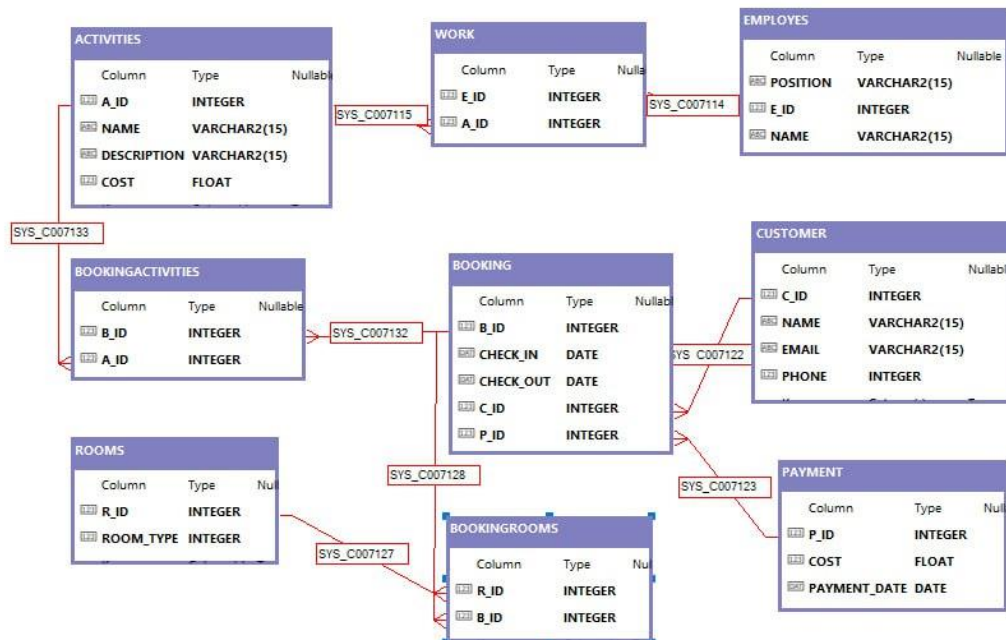
TotalCost-תשלום הזמנה נעשה בהתאם לתכולת ההזמנה.

תרשימים:

ERD:



תרשים DSD:



## Create tables:

```
CREATE TABLE Customer
(
  c_id INT NOT NULL,
  name VARCHAR(15) NOT NULL,
  email VARCHAR(15) NOT NULL,
  phone INT NOT NULL,
  PRIMARY KEY (c_id)
);

CREATE TABLE Rooms
(
  r_id INT NOT NULL,
  room_type INT NOT NULL,
  PRIMARY KEY (r_id)
);

CREATE TABLE Payment
(
  p_id INT NOT NULL,
  cost FLOAT NOT NULL,
  payment_date DATE NOT NULL,
  PRIMARY KEY (p_id)
);

CREATE TABLE Activities
(
  a_id INT NOT NULL,
  name VARCHAR(15) NOT NULL,
  description VARCHAR(15) NOT NULL,
  cost FLOAT NOT NULL,
  PRIMARY KEY (a_id)
);

CREATE TABLE Employees
(
  position VARCHAR(15) NOT NULL,
  e_id INT NOT NULL,
  name VARCHAR(15) NOT NULL,
  PRIMARY KEY (e_id)
);

CREATE TABLE Work
(
  e_id INT NOT NULL,
  a_id INT NOT NULL,
  PRIMARY KEY (e_id, a_id),
  FOREIGN KEY (e_id) REFERENCES Employees(e_id),
  FOREIGN KEY (a_id) REFERENCES Activities(a_id)
);

CREATE TABLE Booking
(
  b_id INT NOT NULL,
  check_in DATE NOT NULL,
  check_out DATE NOT NULL,
  c_id INT NOT NULL,
  p_id INT NOT NULL,
  PRIMARY KEY (b_id),
  FOREIGN KEY (c_id) REFERENCES Customer(c_id),
  FOREIGN KEY (p_id) REFERENCES Payment(p_id)
);

CREATE TABLE BookingRooms
(
  r_id INT NOT NULL,
  b_id INT NOT NULL,
  PRIMARY KEY (r_id, b_id),
  FOREIGN KEY (r_id) REFERENCES Rooms(r_id),
  FOREIGN KEY (b_id) REFERENCES Booking(b_id)
);

CREATE TABLE BookingActivities
(
  b_id INT NOT NULL,
  a_id INT NOT NULL,
  PRIMARY KEY (b_id, a_id),
  FOREIGN KEY (b_id) REFERENCES Booking(b_id),
  FOREIGN KEY (a_id) REFERENCES Activities(a_id)
);
```

## DESC:

Connected to Oracle Database 11g Express Edition Release 11.2.0.2.0  
Connected as hotel@XE

```
SQL> desc customer
Name      Type                Nullable Default Comments
-----
C_ID      INTEGER
NAME      VARCHAR2(15)
EMAIL     VARCHAR2(15)
PHONE     INTEGER
```

```
SQL> desc rooms
Name      Type                Nullable Default Comments
-----
R_ID      INTEGER
ROOM_TYPE INTEGER
```

```
SQL> desc bookingrooms
Name      Type                Nullable Default Comments
-----
R_ID      INTEGER
B_ID      INTEGER
```

```
SQL> desc activities
Name      Type                Nullable Default Comments
-----
A_ID      INTEGER
NAME      VARCHAR2(15)
DESCRIPTION VARCHAR2(15)
COST      FLOAT
```

```
SQL> desc employees
Name      Type                Nullable Default Comments
-----
POSITION  VARCHAR2(15)
E_ID      INTEGER
NAME      VARCHAR2(15)
```

```
SQL> desc payment
Name      Type                Nullable Default Comments
-----
P_ID      INTEGER
COST      FLOAT
PAYMENT_DATE DATE
```

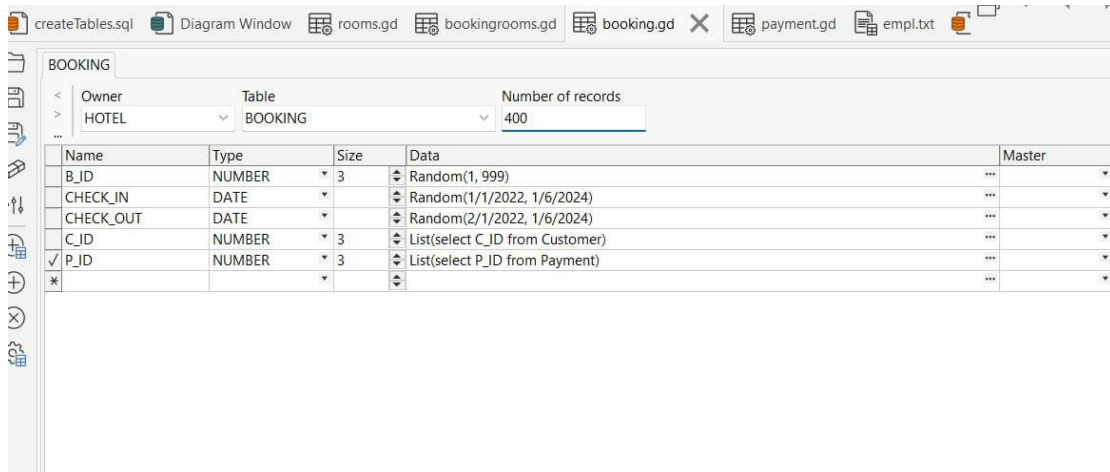
```
SQL> desc bookingactivities
Name      Type                Nullable Default Comments
-----
B_ID      INTEGER
A_ID      INTEGER
```

```
SQL> desc booking
Name      Type                Nullable Default Comments
-----
B_ID      INTEGER
CHECK_IN  DATE
CHECK_OUT DATE
C_ID      INTEGER
P_ID      INTEGER
```

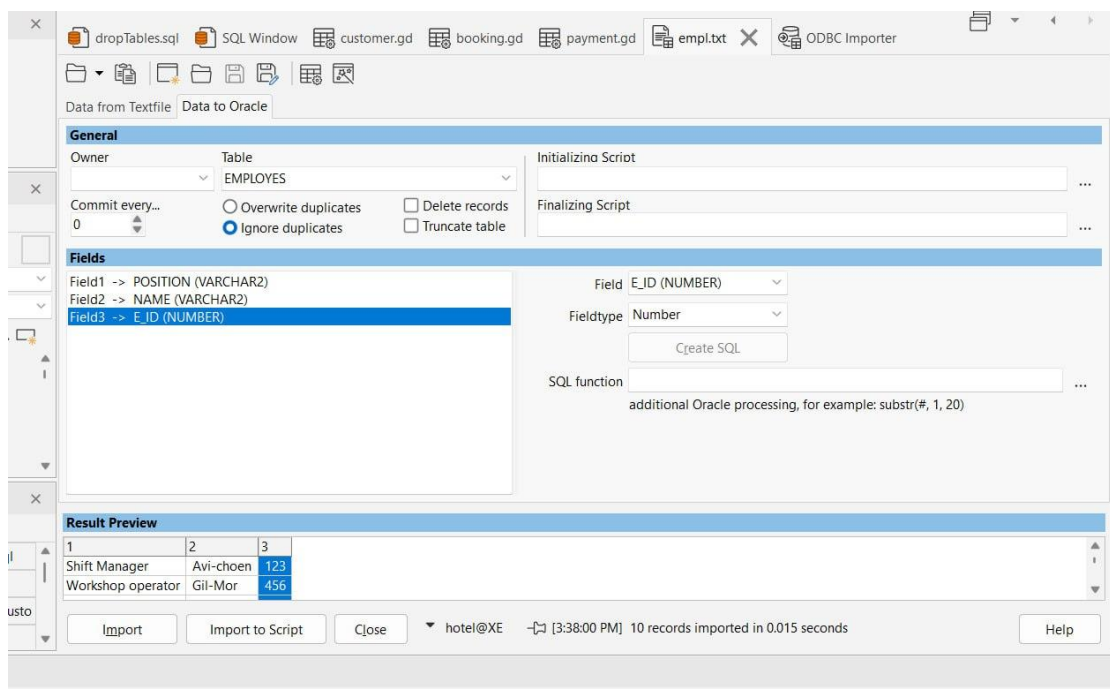
```
SQL> desc work
Name      Type                Nullable Default Comments
-----
E_ID      INTEGER
A_ID      INTEGER
```

## הכנסת נתונים לטבלה בשלושה דרכים שונות:

### 1. data Generator



### 2. קובץ טקסט: DataImporterFile



### 3. אתר אינטרנט: Mockaroo

The Mockaroo website interface shows a schema configuration for a table named 'activities'. The configuration includes the following fields:

Field Name	Type	Options
a_id	Number	min: 1 max: 999 decimals: 0 blank: 0 %
name	Custom List	pool, bar, boating, trip, cars, art, spa, jump, hair style, jip, gymbooree, rooftop part sequential blank: 0 %
description	Custom List	family, adults, family, family, adults, kids, adults, kids, family, adults, kids, adults, s sequential blank: 0 %
cost	Custom List	0, 60, 30, 100, 150, 40, 0, 40, 70, 65, 40, 20, 50 sequential blank: 0 %

Buttons: + ADD ANOTHER FIELD, GENERATE FIELDS USING AI...

# Rows: 13 Format: SQL Table Name: activities include CREATE TABLE

Follow @mockarodev

Mock your back-end API and start coding your UI today.

Buttons: GENERATE DATA, PREVIEW, SAVE AS..., DERIVE FROM EXAMPLE..., MORE

### פעולת הגיבוי:

The Oracle SQL Developer interface shows the 'Export Tables of HOTEL' window. The window displays a list of tables to be exported:

Name	Type	Compiled
ACTIVITIES	TABLE	5/24/2024 1:32:09 PM
BOOKING	TABLE	5/23/2024 2:57:28 PM
BOOKINGACTIVITIES	TABLE	5/24/2024 1:44:40 PM
BOOKINGROOMS	TABLE	5/23/2024 2:57:28 PM
CUSTOMER	TABLE	5/23/2024 2:57:27 PM
EMPLOYES	TABLE	5/23/2024 2:57:27 PM
PAYMENT	TABLE	5/23/2024 2:57:27 PM
ROOMS	TABLE	5/23/2024 2:57:27 PM
WORK	TABLE	5/24/2024 1:44:40 PM

User: <CURRENT USER>

Oracle Export: SQL Inserts PL/SQL Developer

Options:

- ☐ Drop tables
- ☒ Create tables
- ☐ Truncate tables
- ☒ Delete records
- ☒ Disable triggers
- ☒ Disable foreign key constraints
- ☒ Include storage
- ☒ Include privileges

Commit every 100 records (0 = never)

☐ Zip Where clause




Output file: C:\Users\lino\OneDrive\העבודה\שולחן\database\backup24\_5\_24.sql

Buttons: Export

hotel@XE



## פעולת שחזור הנתונים:

Oracle Import	SQL Inserts	PL/SQL Developer	Log
<div><input type="checkbox"/> Use Command Window</div> <div><input checked="" type="checkbox"/> Use SQL*Plus</div> <div>SQL*Plus Executable</div> <div>C:\oracle\app\oracle\product\11.2.0\server\bin\sqlplus.exe  </div>			
<div>Import file</div> <div>C:\Users\linoy\OneDrive\העבודה\שולחן העבודה\database\שלב א\backup24_5.sql </div> <div>Import</div>			

# שאלות

## ארבעה שאלות:

1. השאלה מחזירה את שמות הלקוחות, מספר הפעילויות שלקחו ואת הסכום הכללי של כל הפעילויות יחד, אך רק כאלו שלקחו יותר מפעילות אחת בהזמנה.

```
SELECT name AS customer_name, num_of_activities, activities_cost
FROM (SELECT c.c_id, c.name, count(*) AS num_of_activities, sum(a.cost) AS activities_cost
      FROM Customer c JOIN Booking b ON c.c_id = b.c_id JOIN BookingActivities ba ON b.b_id = ba.b_id
      JOIN Activities a ON ba.a_id = a.a_id
      GROUP BY c.c_id, c.name
      HAVING COUNT(*) > 1)
ORDER BY activities_cost DESC;
```

	CUSTOMER_NAME	NUM_OF_ACTIVITIES	ACTIVITIES_COST
1	Jarvis-Berkeley	5	340
2	Ronny-Rhymes	3	240
3	Kay-Griffin	4	210
4	Scarlett-Dysart	2	110
5	Ossie-Cazale	2	100

28:6 hotel@XE [10:23:20 PM] 5 rows selected in 0.025 seconds

2. השאלה מחזירה את פרטי הלקוחות שלקחו את כל ארבעת סוגי החדרים השונים שיש למלון להציע.

```
SELECT * --מחזיר את הלקוחות שהזמינו את כל סוגי החדרים
FROM customer c
WHERE NOT EXISTS (SELECT DISTINCT(r.room_type) --מחזיר את כל סוגי החדרים שכל לקוח לא שכר
                  FROM rooms r
                  WHERE r.room_type NOT IN (SELECT DISTINCT(r1.room_type) --מחזיר את כל סוגי החדרים של כל לקוח
                                           FROM bookingrooms br NATURAL JOIN rooms r1 NATURAL JOIN booking b
                                           WHERE b.c_id = c.c_id));
```

	C.ID	NAME	EMAIL	PHONE
1	232	Maria-Lightfoot	maria@serentec.	588946848
2	553	Sigourney-Hauer	Sigourney@com	579503319
3	678	Catherine-Vance	catherine@v.il	573633070
4	639	Nicky-Bush	nicky.bush@uem.	593206710
5	88	Terence-Neuwirt	Terence@com	501507763

95:89 hotel@XE [10:40:36 PM] 5 rows selected in 0.124 seconds

3. השאילתה מחזירה את שמות העובדים ותפקידם העיקרי, אך רק את העובדים שעובדים בכל אחד מסוג הפעילויות שיש למלון להציע לאורחים.

```

SELECT e.name AS employee_name, e.position AS main_position -- מחזיר את שמות העובדים ותפקידם העיקרי ששייכים לכל הפעילויות
FROM employees e JOIN work w ON e.e_id = w.e_id
WHERE w.a_id NOT IN (SELECT a.a_id -- מחזיר את הפעילויות שהעובד לא שייך אליהם
                     FROM activities a
                     WHERE NOT EXISTS (SELECT w2.a_id -- מחזיר את הפעילויות של העובד
                                       FROM work w2
                                       WHERE w2.a_id = a.a_id AND w2.e_id = e.e_id))
GROUP BY e.name, e.position;

```

	EMPLOYEE_NAME
1	Noa-Sir
2	Saar-Roi
3	Noga-Shir
4	Noa-Gor
5	Avi-choen
6	Ron-Cali

155:99 hotel@XE [10:59:36 PM] 6 rows selected in 0.043 seconds (more...)

4. השאילתה מחזירה את שמות הלקוחות, את הסכום הכולל שלהם בהזמנה ואת תאריך התשלום, אך רק את אלו שסכום התשלום שלהם הוא נע בין 1000 ל3000 ותאריך התשלום הוא אחרי תאריך עזיבתם את המלון.

```

SELECT c.name AS customer_name, p.cost, p.payment_date -- ותאריך התשלום שלהם הוא אחרי תאריך היציאה מהמלון
FROM Customer c JOIN Booking b ON c.c_id = b.c_id JOIN Payment p ON b.p_id = p.p_id
WHERE (c.c_id, p.p_id) IN (SELECT b.c_id, p1.p_id -- לקוחות ששילמו יותר מ1000
                          FROM booking b, payment p1
                          WHERE b.p_id = p1.p_id and p1.cost > 1000
                          MINUS
                          SELECT b1.c_id, p2.p_id -- לקוחות ששילמו יותר מ3000
                          FROM booking b1, payment p2
                          WHERE b1.p_id = p2.p_id and p2.cost > 3000) AND p.payment_date > b.check_out
ORDER BY p.cost ASC;

```

	CUSTOMER_NAME	COST	PAYMENT_DATE
1	Solomon-McLachl	1065	10/22/2023
2	Ming-Na-Warwick	1065	10/22/2023
3	Remy-Curtis-Hal	1081	5/11/2022
4	Ethan-Malkovich	1092	10/6/2022
5	Stellan-Wine	1092	10/6/2022
6	Kay-Griffin	1118	8/4/2023

100:1 hotel@XE [11:21:07 PM] 62 rows selected in 0.106 seconds

## ארבעה שאילות עם פרמטרים:

1. השאילתה מחזירה את שם העובד, את התפקיד שלו ואת סוג הפעילות בה הוא מעורב בהתאם לפרמטרים שהוכנסו של סוג הפעילות והתפקיד הנצרך.

מזהה עובד ופעילות ומציג את שם העובד, התפקיד שלו והפעילות שבה הוא מעורב--

```
SELECT e.name AS employee_name, position, a.name AS activity_name
FROM Work w JOIN Employees e ON w.e_id = e.e_id JOIN Activities a ON w.a_id = a.a_id
WHERE e.position = <name="position"
      type="string"
      list="SELECT distinct position FROM Employees">
AND w.a_id = <name="Activity"
      type="string"
      list="SELECT a.a_id, a.name FROM Activities a ORDER BY a_id" description="true" restricted="true">;
```

Variables

Name	Value
position	Driver
Activity	jip

OK Cancel Clear

	EMPLOYEE_NAME	POSITION	ACTIVITY_NAME
1	Noa-Sir	Driver	jip
2	Amir-Lev	Driver	jip

2. השאילתה מחזירה את שמות האורחים, סוג החדר הנבחר בשנה מסוימת שנבחרה ואת כמותו, אך רק את אלו שסך החדרים שלקחו מכל הסוגים שווה למספר שנבחר.

הלקוחות שביצעו חדרים שלקחו בשנה שנבחרה שווה למספר שהוכנס וגם שאחד החדרים הוא מסוג החדר שהוכנס ויחזיר רק אלוהם שיש חדר שהוכנס וכמה לקחו מסוג זה

```
SELECT c.name AS customer, r.room_type, count(*) AS number_of_rooms
FROM Customer c JOIN Booking b ON c.c_id = b.c_id JOIN BookingRooms br ON b.b_id = br.b_id JOIN Rooms r ON br.r_id = r.r_id
WHERE (c.c_id IN (SELECT c1.c_id
FROM Customer c1 JOIN Booking b1 ON c1.c_id = b1.c_id JOIN BookingRooms br1 ON b1.b_id = br1.b_id
GROUP BY c1.c_id, b1.b_id
HAVING COUNT(*) = <name="number of rooms"
      type="integer"
      required="true">)) AND r.room_type = <name="rooms type"
      type="integer"
      list="SELECT DISTINCT room_type FROM Rooms ORDER BY room_type"
      default="2"
      hint="room_type means number of beds per room which is between 1-5">
AND EXTRACT(YEAR FROM TO_DATE(b.check_in, 'DD/MM/YYYY')) = <name="year"
      type="integer"
      list="22, 23, 24"
      restricted="yes">
GROUP BY c.c_id, r.room_type, c.name;
```

Variables

Name	Value
number of rooms	3
rooms type	3
year	22

OK Cancel Clear

room\_type means number of beds per room which is between 1-5

	CUSTOMER	ROOM_TYPE	NUMBER_OF_ROOMS
1	Debbie-Buscemi	3	2

3. השאילתה מחזירה את שמות האורחים, הפעילות, למי היא מיועדת ותאריך הכניסה למלון, של אורחים שהזמינו פעילות המיועדת לקהל מסוים שנבחר על ידי המשתמש בשנה וחצי האחרונות.

The screenshot shows a SQL query in a blue editor window. The query is as follows:

```
SELECT c.name, a.name, a.description, b.check_in
FROM customer c JOIN booking b ON c.c_id=b.c_id JOIN BOOKINGACTIVITIES ba ON b.b_id=ba.b_id JOIN activities a ON ba.a_id=a.a_id
WHERE a.description IN (<name="activities for:"
    type="string"
    list="adults, kids, family"
    multiselect="yes"
    hint="Specifies the target audience for which you are looking for an activity">)
AND b.check_in BETWEEN ADD_MONTHS(TRUNC(SYSDATE), -18) AND TRUNC(SYSDATE)
ORDER BY b.check_in DESC;
```

Below the query is a table with 6 rows and 4 columns: NAME, DESCRIPTION, CHECK\_IN, and an empty column. The data is as follows:

	NAME	DESCRIPTION	CHECK_IN	
1	Vickie-Hanley	cars	family	8/6/2023
2	Sigourney-Hauer	art	kids	6/7/2023
3	Vonda-Owen	trip	family	2/24/2023
4	Don-Johnson	art	kids	2/20/2023
5	Domingo-Feore	jump	kids	2/2/2023
6	Debbie-Buscemi	art	kids	12/24/2022

On the right, there is a 'Select values' dialog box with a list of activities: adults, kids, and family. The 'kids' and 'family' options are selected. Below the list is a 'Variables' section with a table:

Name	Value
activities for:	kids, family

Below the table is a text label: 'Specifies the target audience for which you'.

4. השאילתה מחזירה את שמות הלקוחות, הת.ז. שלהם סוג החדר שלקחו בהתאם לסוגי החדרים שנבחרו, ואת תאריך התשלום אך רק לאלו שהתאריך תשלום שלהם הוא אחרי התאריך שהוזן על ידי המשתמש.

The screenshot shows a SQL query in a blue editor window. The query is as follows:

```
Select C.c_id, C.name, R.room_type, P.payment_date
From CUSTOMER C JOIN BOOKING B on C.C_id = B.C_id JOIN BOOKINGROOMS BR on BR.B_id = B.B_id JOIN
ROOMS R on BR.R_id = R.R_id JOIN PAYMENT P ON B.P_ID=P.P_ID
Where R.room_type in ('&room_type1', '&room_type2')
and P.payment_date > TO_DATE('&payment_date', 'DD/MM/YYYY')
Group by C.C_ID, C.NAME, R.room_type, P.payment_date
ORDER BY c.name <name="room" checkbox="desc, asc">;
```

Below the query is a table with 7 rows and 4 columns: C\_ID, NAME, ROOM\_TYPE, and PAYMENT\_DATE. The data is as follows:

	C_ID	NAME	ROOM_TYPE	PAYMENT_DATE
1	143	Alex-Sanders	2	4/1/2023
2	281	Andre-Crewson	1	4/26/2023
3	737	Brittany-Hawtho	2	3/20/2023
4	334	Chi-Levin	1	4/28/2023
5	334	Chi-Levin	2	5/27/2023
6	699	Collin-Rollins	1	3/7/2023
7	90	Darius-LuPone	1	1/16/2023

On the right, there is a 'Variables' dialog box with a table:

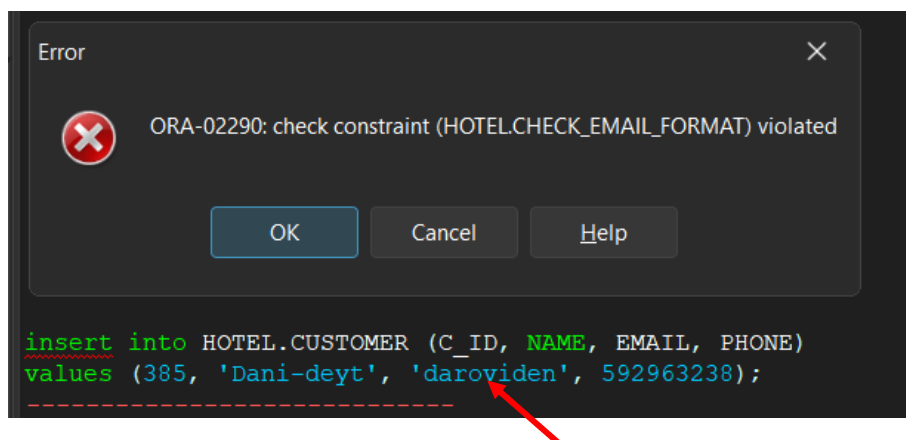
Name	Value
room_type1	2
room_type2	1
payment_date	04/01/2023
room	

## אילוצים:

1. האילוץ בודק שכתובת האימייל הוכנסה כראוי, כלומר שב-@ נכתב כחלק מכתובת מייל תקינה.

```
ALTER TABLE Customer
ADD CONSTRAINT check_email_format CHECK (INSTR(email, '@') > 0);
```

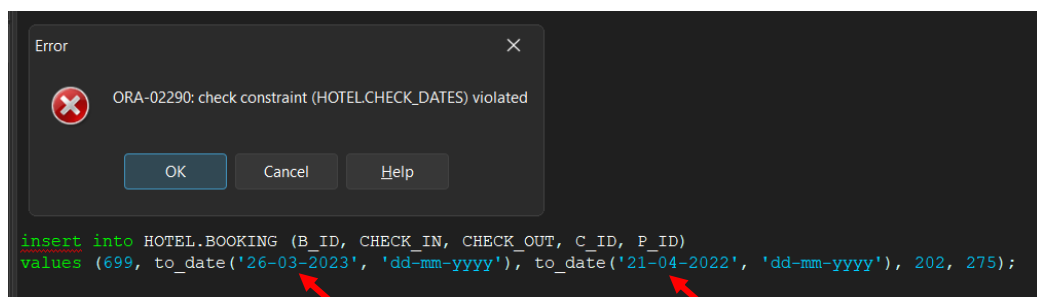
נראה דוגמא של הכנסת מייל לא תקינה ובכך נוצר שגיאת ריצה:



2. האילוץ בודק שתאריך הכניסה למלון הוא אכן לפני תאריך עזיבת המלון.

```
ALTER TABLE Booking
ADD CONSTRAINT check_dates
CHECK (check_in < check_out);
```

נראה דוגמא של הכנסת תאריכים לא תקינה ויינוצר שגיאת ריצה:



3. אילוח של ברירת מחדל, כאשר לא הכניסו תפקיד מסוים לעובד, התפקיד שינתן לו אוטומטית יהיה עובד דלפק.

```
ALTER TABLE Employees
MODIFY position DEFAULT 'Counter Clerk';
```

נראה דוגמא להכנסת עובד חדש ללא כתיבת תפקידו:

```
insert into employees ("NAME", "E_ID")
values ('Noga-Shir', 75);

select * from employees where e_id = 75;
```

	POSITION	E_ID	NAME
1	Counter Clerk	75	Noga-Shir

## שאלות עידכון:

1. השאילתה מעדכנת את טבלת התשלום כך שהתשלום יוזל ב100 שקלים לכל לקוח שסכום התשלום הכולל שלו עולה על הסכום הממוצע ששילמו לקוחות בשנת 2022, וגם שמספר הפעילויות שהזמין עולה על מספר הפעילויות הממוצע שהזמינו לקוחות באותה שנה.

```
UPDATE PAYMENT P
SET p.cost = p.cost-100
WHERE p.p_id IN (SELECT p2.p_id FROM payment p2 JOIN booking b1 ON p2.p_id = b1.p_id WHERE p2.p_id IN
(SELECT p1.p_id
FROM payment p1
WHERE p1.cost > (SELECT AVG(p2.cost)
FROM payment p2
WHERE p2.Payment_Date LIKE '%_2') AND b1.c_id IN (SELECT b.c_id
FROM Booking b JOIN BookingActivities ba ON b.b_id = ba.b_id
GROUP BY b.c_id
HAVING COUNT(ba.a_id) > (SELECT AVG(activity_count) AS activity_avg
FROM (SELECT COUNT(ba.a_id) AS activity_count
FROM Booking b JOIN BookingActivities ba ON b.b_id = ba.b_id
WHERE b.check_in LIKE '%_2'
GROUP BY b.c_id))));
```

בסיס הנתונים לפני עידכון:

	P_ID	COST	PAYMENT_DATE	
▶ 1	110	80929	1/7/2022	...
2	213	64613	2/14/2022	...
3	310	95123	3/14/2022	...

בסיס הנתונים אחרי עידכון המחירים:

	P_ID	COST	PAYMENT_DATE	
▶ 1	110	80829	1/7/2022	...
2	213	64513	2/14/2022	...
3	310	95023	3/14/2022	...

2. השאילתה מעדכנת את טבלת החדרים בכך שמוסיפה לטבלה עמודה חדשה המציינת את קומת החדר במלון:

```
ALTER TABLE Rooms
ADD floor INT;

BEGIN
  FOR i IN 2..20 LOOP
    UPDATE Rooms
    SET floor = i
    WHERE r_id IN (
      SELECT r_id
      FROM (
        SELECT r_id
        FROM Rooms
        WHERE floor IS NULL
        ORDER BY DBMS_RANDOM.VALUE
      )
      WHERE ROWNUM <= 40
    );
  END LOOP;
END;
```

השתמשנו בפונקציה על מנת למלא את השורות בעמודה החדשה שיצרנו בהתאם.



## שאלות מחיקה:

1. השאלתה מוחקת את כל החדרים מרשימת החדרים שהוזמנו, את אלו שיש להם סוג חדר של 3 מיטות, וגם לא מכילות חדרים מסוג 4,5.

```
-- מחיקת הרשומות מהטבלה המקורית --
DELETE FROM BookingRooms
WHERE r_id IN (SELECT br1.r_id
               FROM BookingRooms br1 JOIN Rooms r1 ON br1.r_id = r1.r_id
               WHERE r1.room_type = 3 AND br1.b_id NOT IN (SELECT br2.b_id
                                                           FROM BookingRooms br2 JOIN Rooms r2 ON br2.r_id = r2.r_id
                                                           WHERE r2.room_type IN (4, 5)));
```

הטבלה לפני המחיקה עם ההזמנות המכילות חדרים מסוג 3 ולא חדרים מסוג 4,5 באותה הזמנה.

```
SELECT b_id, count(*) FROM BookingRooms
WHERE r_id IN (SELECT br1.r_id
               FROM BookingRooms br1 JOIN Rooms r1 ON br1.r_id = r1.r_id
               WHERE r1.room_type = 3 AND br1.b_id NOT IN (SELECT br2.b_id
                                                           FROM BookingRooms br2 JOIN Rooms r2 ON br2.r_id = r2.r_id
                                                           WHERE r2.room_type IN (4, 5)))
GROUP BY BookingRooms.b_id;
```

	B_ID	COUNT(*)
1	123	1
2	311	1
3	455	1
4	484	1
5	407	1
6	100	1
7	555	1
8	670	1
9	696	1
10	217	1
11	353	1
12	534	1
13	543	1
14	657	2
15	262	1
16	897	1

hotel@XE [12:40:31 PM] 45 rows selected in 0.105 seconds

הטבלה אחרי המחיקה מסוננת בהתאם לתנאי המחיקה, לא נראה את ההזמנות שמכילות חדרים מסוג 3 ולא מכילות חדרים מסוג 4 או 5 באותה הזמנה.

```
SQL Output Statistics
SELECT b_id, count(*) FROM BookingRooms
WHERE r_id IN (SELECT br1.r_id
               FROM BookingRooms br1 JOIN Rooms r1 ON br1.r_id = r1.r_id
               WHERE r1.room_type = 3 AND br1.b_id NOT IN (SELECT br2.b_id
                                                           FROM BookingRooms br2 JOIN Rooms r2 ON br2.r_id = r2.r_id
                                                           WHERE r2.room_type IN (4, 5)))
GROUP BY BookingRooms.b_id;
```

	B_ID	COUNT(*)
--	------	----------

2. השאילתה מוחקת את כל ההזמנות שזמן ימי האירוח שלהם במלון הוא לכל היותר שבעה ימים, ומספר הפלאפון של הלקוח המזמין מסתיים בספרה 1.

```
DELETE FROM Booking
WHERE check_in <= check_out - 7 AND booking.c_id IN (SELECT c_id FROM Customer
WHERE phone LIKE '%_1');

DELETE FROM BookingRooms
WHERE b_id IN (SELECT b_id FROM Booking
WHERE check_in <= check_out - 7 AND booking.c_id IN (SELECT c_id FROM Customer
WHERE phone LIKE '%_1'));

DELETE FROM BookingActivities
WHERE b_id IN (SELECT b_id FROM Booking
WHERE check_in <= check_out - 7 AND booking.c_id IN (SELECT c_id FROM Customer
WHERE phone LIKE '%_1'));
```

נשים לב כיוון שהמפתח בטבלת ההזמנות משמש כמפתח זר בטבלאות אחרות, נצטרך למחוק גם מהטבלאות המקושרות את הרשומות המתאימות.

הטבלה של ההזמנות לפני מחיקת הנתונים הרצויים.

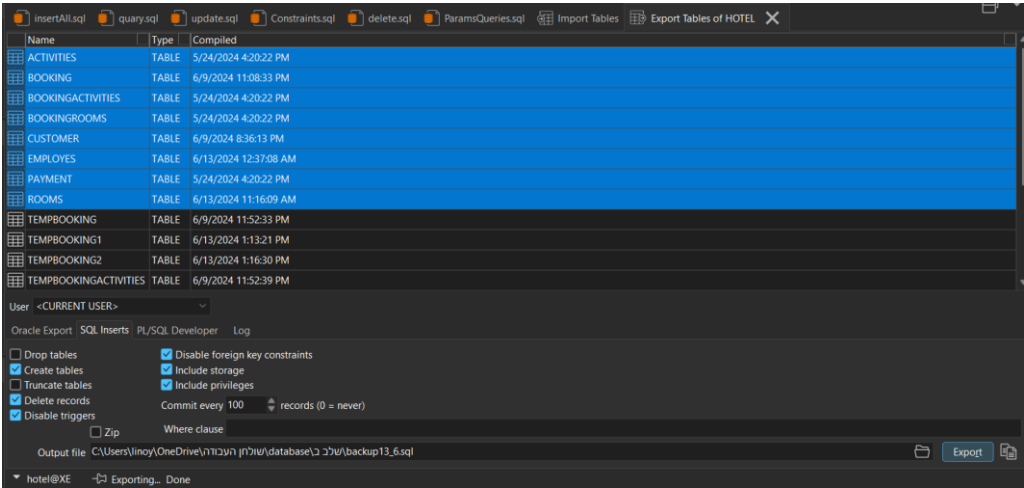
```
SELECT * FROM Booking
WHERE check_in <= check_out - 7 AND booking.c_id IN (SELECT c_id FROM Customer
WHERE phone LIKE '%_1');
```

	B_ID	CHECK_IN	CHECK_OUT	C_ID	P_ID
1	72	5/9/2022	6/23/2023	90	614
2	469	11/14/2022	1/1/2024	942	387
3	23	8/28/2022	8/26/2023	246	670
4	356	9/26/2022	6/14/2023	442	699
5	359	4/16/2022	12/1/2023	695	744
6	545	5/17/2023	12/28/2023	415	292
7	651	12/21/2022	2/14/2023	246	496
8	971	8/6/2022	11/21/2023	122	674
9	492	4/14/2022	4/17/2023	527	901
10	89	3/12/2022	5/13/2022	695	461
11	160	9/3/2022	12/9/2022	246	847
12	699	5/15/2022	6/4/2022	583	213
13	490	12/20/2022	4/6/2023	449	633
14	675	3/20/2022	9/12/2023	56	39
15	316	5/16/2022	5/30/2023	56	62
16	904	8/12/2022	1/4/2024	396	937
17	543	7/29/2023	8/15/2023	534	596

הטבלה של ההזמנות אחרי המחיקה, מסוננת לפי תנאי המחיקה ולכן נקבל טבלה ריקה.

```
SELECT * FROM Booking
WHERE check_in <= check_out - 7 AND booking.c_id IN (SELECT c_id FROM Customer
WHERE phone LIKE '%1');
```

גיבוי 2:



## תוכניות

### תוכנית 1:

התוכנית בודקת מי הם כל הלקוחות שהזמינו פעילות העולה על 70 שח ונותנת להם פעילות נוספת במתנה.

תוכנית ראשית - התוכנית בודקת על ידי שאילתה מי הם אותם לקוחות ועל ידי לולאה שולחת בכל פעם מספר לקוח לפרוצדורה אשר שם מוסיפה לו פעילות רנדומלית.

```
1 DECLARE
2   CURSOR customersToActivate IS
3     SELECT DISTINCT b.c_id
4     FROM Booking b
5     WHERE b.b_id IN (SELECT ba.b_id
6                      FROM BookingActivities ba JOIN activities a ON ba.a_id=a.a_id
7                      WHERE a.cost > 70); -- Assuming get_customers() returns a nested table or varray
8   new_booking_activities_cnt NUMBER := 0;
9   cust_rec customer.c_id%TYPE;
10 BEGIN
11   OPEN customersToActivate;
12   LOOP
13     exit when customersToActivate%notfound;
14     fetch customersToActivate into cust_rec;
15     -- Call procedure to add random activity for each customer
16     add_random_activity(cust_rec);
17     new_booking_activities_cnt := new_booking_activities_cnt + SQL%ROWCOUNT;
18   END LOOP;
19   CLOSE customersToActivate;
20   DBMS_OUTPUT.PUT_LINE('Total new rows added: ' || new_booking_activities_cnt);
21 EXCEPTION
22   WHEN OTHERS THEN
23     ROLLBACK;
24     DBMS_OUTPUT.PUT_LINE('An error occurred: ' || SQLERRM);
25 END;
```

פרוצדורה – מקבלת מהתוכנית הראשית מספר לקוח, ובהתאם לקח שולחת לפונקציה שמחזירה את מספר ההזמנה, ולפונקציה נוספת שמחזירה רנדומלית אחת מהפעילויות שיש למלון להציע, ומוסיפה לטבלה של הזמנות פעילות את הפעילות שהתקבלה ואת מספר ההזמנה של אותו לקוח, ומדפיסה את הנתונים.

```
1 CREATE OR REPLACE PROCEDURE add_random_activity(cust_id IN NUMBER)
2 IS
3   activity_id activities.a_id%TYPE;
4   booking_id booking.b_id%TYPE;
5 BEGIN
6   booking_id := booking_value(cust_id);
7
8   -- Select a random activity_id from activities table
9   activity_id := add_act;
10  IF activity_id IS NOT NULL THEN
11
12    -- Insert into BookingActivities
13    INSERT INTO BookingActivities (b_id, a_id)
14    VALUES (booking_id, activity_id);
15
16    -- Print the added activity for customer
17    DBMS_OUTPUT.PUT_LINE('Added activity ' || activity_id || ' for customer ' || cust_id);
18  ELSE
19    DBMS_OUTPUT.PUT_LINE('No activities available to add for customer ' || cust_id);
20  END IF;
21 EXCEPTION
22   WHEN OTHERS THEN
23     ROLLBACK;
24     DBMS_OUTPUT.PUT_LINE('An error occurred: ' || SQLERRM);
25 END add_random_activity;
```

פונקציה 1 – מחזירה לפרוצדורה את מספר הזמנה של לקוח שמקבלת.

```
create or replace function booking_value(cust_id IN NUMBER)
return number is
    FunctionResult number;
begin
    SELECT b_id INTO FunctionResult
    FROM booking
    WHERE c_id = cust_id
    AND ROWNUM = 1
    ORDER BY DBMS_RANDOM.VALUE;

    return(FunctionResult);
end booking_value;
```

פונקציה 2 – מחזירה לפרוצדורה מספר פעילות מסוימת מתוך הפעילויות של המלון.

```
1 create or replace function add_act
2 RETURN activities.a_id%TYPE IS
3     activity_id activities.a_id%TYPE;
4 BEGIN
5     -- Select a random activity_id from activities table
6     SELECT a_id INTO activity_id
7     FROM (SELECT a_id FROM activities ORDER BY DBMS_RANDOM.VALUE)
8     WHERE ROWNUM = 1;
9
10    RETURN activity_id;
11 EXCEPTION
12 WHEN NO_DATA_FOUND THEN
13     RETURN NULL; -- Return NULL if no activities found
14 WHEN OTHERS THEN
15     RAISE;
16 end add_act;
```

פלט: ניתן לראות ש15 אורחים עמדו בתנאי וקיבלו פעילות מתנה כל אחד מהם בצורה רנדומלית.

```
Added activity 389 for customer 66
Added activity 78 for customer 326
Added activity 759 for customer 759
Added activity 527 for customer 274
Added activity 824 for customer 170
Added activity 502 for customer 709
Added activity 298 for customer 219
Added activity 45 for customer 673
Added activity 78 for customer 747
Added activity 759 for customer 863
Added activity 596 for customer 327
Added activity 794 for customer 931
Added activity 389 for customer 318
Added activity 45 for customer 215
Added activity 392 for customer 215
Total new rows added: 15
```

## תוכנית 2:

התוכנית מקבלת מספר לקוח ובהתאם להזמנות של אותו לקוח ואיזה פעילויות הוא לקח, היא נותנת 10 אחוז תוספת במשכורת לאותם עובדים שעסקו באותם פעילויות וכן דואגת להעלות גם את המחיר שישלם אותו לקוח, בעצם הלקוח שילם טיפ של 10 אחוז לעובדים.

תוכנית ראשית – מקבלת מפונקציה בצורה רנדומלית מספר לקוח ושולחת לפונקציה אחרת על מנת לקבל את פרטי ההזמנה של אותו לקוח ובכך שולחת את מספר ההזמנה (יכול להיות כמה הזמנות לאותו לקוח) לפונקציה נוספת שמחזירה את כל הפעילויות ששכר הלקוח ומעדכנת בפרוצדורה את משכורת העובדים ואת מחיר התשלום של הלקוח בכל הזמנה.

```
DECLARE
    bookingsCursor SYS_REFCURSOR;
    activitiesCursor SYS_REFCURSOR;
    b_id Booking.b_id%TYPE;
    c_id Booking.c_id%TYPE;
    check_in Booking.check_in%TYPE;
    check_out Booking.check_out%TYPE;
    a_id Activities.a_id%TYPE;
    a_name Activities.name%TYPE;
    p_c_id customer.c_id%TYPE;

BEGIN
    p_c_id := get_id;
    -- Call the function to get the bookings for the customer
    bookingsCursor := get_customer_bookings(p_c_id);

    -- Loop through the cursor and display the booking details
    LOOP
        FETCH bookingsCursor INTO b_id, c_id, check_in, check_out;
        EXIT WHEN bookingsCursor%NOTFOUND;

        activitiesCursor := get_booking_activities(b_id);

        -- Loop through the activities cursor and display the activities details
        LOOP
            FETCH activitiesCursor INTO a_id, a_name;
            EXIT WHEN activitiesCursor%NOTFOUND;

            update_salary(a_id, b_id);

            DBMS_OUTPUT.PUT_LINE('Activity ID: ' || a_id || ', Activity Name: ' || a_name);
        END LOOP;
        CLOSE activitiesCursor;
    END LOOP;
    CLOSE bookingsCursor;
EXCEPTION
    WHEN OTHERS THEN
        DBMS_OUTPUT.PUT_LINE('An error occurred: ' || SQLERRM);
END;
```

פונקציה 1 – מחזירה בצורה רנדומלית לתוכנית הראשית את מספר הלקוח.

```
create or replace function get_id
return number is
    FunctionResult number;
begin
    SELECT c_id INTO FunctionResult
    FROM (
        SELECT c_id
        FROM customer
        ORDER BY DBMS_RANDOM.VALUE
    )
    WHERE ROWNUM = 1;

    return(FunctionResult);
end get_id;
```

פונקציה 2 – מקבלת מספר לקוח ומחזירה פרטי הזמנות של אותו לקוח.

```
CREATE OR REPLACE FUNCTION get_customer_bookings(p_c_id IN NUMBER)
RETURN SYS_REFCURSOR IS
    bookingsCursor SYS_REFCURSOR;
    b_id Booking.b_id%TYPE;
    c_id Booking.c_id%TYPE;
    check_in Booking.check_in%TYPE;
    check_out Booking.check_out%TYPE;
BEGIN
    OPEN bookingsCursor FOR
        SELECT b.b_id, b.c_id, b.check_in, b.check_out
        FROM Booking b
        WHERE b.c_id = p_c_id;
    LOOP
        FETCH bookingsCursor INTO b_id, c_id, check_in, check_out;
        EXIT WHEN bookingsCursor%NOTFOUND;

        DBMS_OUTPUT.PUT_LINE('Booking ID: ' || b_id || ', Customer ID: ' || c_id || ', Check-in: '
        || check_in || ', Check-out: ' || check_out);
    END LOOP;
    CLOSE bookingsCursor;
    OPEN bookingsCursor FOR
        SELECT b.b_id, b.c_id, b.check_in, b.check_out
        FROM Booking b
        WHERE b.c_id = p_c_id;
    RETURN bookingsCursor;
END get_customer_bookings;
```

פונקציה 3 – מקבלת מספר הזמנה ומחזירה את הפעילויות שנשכרו בהזמנה זו.

```
CREATE OR REPLACE FUNCTION get_booking_activities(p_b_id IN NUMBER)
RETURN SYS_REFCURSOR IS
    activitiesCursor SYS_REFCURSOR;
BEGIN
    DBMS_OUTPUT.PUT_LINE('Booking ID: ' || p_b_id);

    OPEN activitiesCursor FOR
        SELECT a.a_id, a.name
        FROM BookingActivities ba JOIN Activities a ON ba.a_id = a.a_id
        WHERE ba.b_id = p_b_id;

    RETURN activitiesCursor;
END get_booking_activities;
```

פרוצדורה – מעדכנת את משכורת העובדים על ידי שמקבלת מספר פעילות ובודקת מי הם אותם עובדים שעוסקים בפעילות זו ומעדכנת את המשכורת שלהם ב10 אחוז.

בנוסף מקבלת מספר הזמנה ומעדכנת את תשלום ההזמנה בהתאם ל10 אחוז של אותו עובד ששילמו לו.

```
CREATE OR REPLACE PROCEDURE update_salary(p_a_id IN NUMBER, p_b_id IN NUMBER)
IS
    CURSOR employeeCursor IS
        SELECT e.e_id, e.salary
        FROM Employees e
        JOIN work w ON e.e_id = w.e_id
        JOIN Activities a ON w.a_id = a.a_id
        WHERE a.a_id = p_a_id;

    payment_update payment.cost%TYPE;
    employee_id Employees.e_id%TYPE;
    current_salary Employees.salary%TYPE;
    temp NUMBER;
BEGIN
    SELECT p.cost INTO payment_update
    FROM booking b JOIN payment p ON b.p_id=p.p_id
    WHERE b.b_id=p_b_id;

    IF payment_update IS NOT NULL THEN
        -- Loop through each employee associated with the activity
        FOR employee_rec IN employeeCursor LOOP
            employee_id := employee_rec.e_id;
            current_salary := employee_rec.salary;

            -- Update the employee's salary
            UPDATE Employees
            SET salary = current_salary * 1.10
            WHERE e_id = employee_id;

            DBMS_OUTPUT.PUT_LINE('Updated salary for employee ' || employee_id || ' for ' || current_salary * 1.10);
            -- Update the customer's payment
            UPDATE payment
            SET cost = cost + current_salary * 0.10
            WHERE p_id IN (SELECT p.p_id
                        FROM booking b JOIN payment p ON b.p_id=p.p_id
                        WHERE b.b_id=p_b_id);

            temp := payment_update + current_salary * 0.10;
            DBMS_OUTPUT.PUT_LINE('Updated payment' || p_b_id || ' for ' || temp);
        END LOOP;
        COMMIT; -- Commit the transaction
    END IF;

EXCEPTION
    WHEN NO_DATA_FOUND THEN
        DBMS_OUTPUT.PUT_LINE('No employee found for activity ' || p_a_id);
    WHEN OTHERS THEN
        ROLLBACK;
        DBMS_OUTPUT.PUT_LINE('An error occurred: ' || SQLERRM);
END update_salary;
```

**פלט:** ניתן לראות שהתקבל לקוח בעל שני הזמנה, משכורת שני העובדים שעוסקים בפעילות שהלקוח לקח התעדכנו, וכן מחיר ההזמנה שלו התעדכן גם כן.

```
Booking ID: 403, Customer ID: 175, Check-in: 04-SEP-22, Check-out: 05-SEP-22
Booking ID: 662, Customer ID: 175, Check-in: 24-JAN-22, Check-out: 02-JUN-22
Booking ID: 403
Activity ID: 20, Activity Name: jip
Updated salary for employee 155 for 4177.248169415651
Updated salary for employee 125 for 4177.248169415651
Updated payment 403 for 6475.749833583241
Booking ID: 662
Activity ID: 392, Activity Name: gymboree
Updated salary for employee 748 for 1331
Updated payment 662 for 7863
|
```



### גיבוי 3:

get\_customer\_bookings.fncmain\_program2.fncTest Windowupdate\_salary.prcget\_booking\_activities.fncgetLid.fncExport Tables of HOTEL

Name	Type	Compiled
ACTIVITIES	TABLE	5/24/2024 4:20:22 PM
BOOKING	TABLE	6/9/2024 11:08:33 PM
BOOKINGACTIVITIES	TABLE	5/24/2024 4:20:22 PM
BOOKINGROOMS	TABLE	5/24/2024 4:20:22 PM
CUSTOMER	TABLE	6/9/2024 8:36:13 PM
EMPLOYES	TABLE	6/21/2024 6:00:36 PM
PAYMENT	TABLE	5/24/2024 4:20:22 PM
ROOMS	TABLE	6/13/2024 11:16:09 AM
TEMPBOOKING	TABLE	6/9/2024 11:52:33 PM
TEMPBOOKING1	TABLE	6/13/2024 1:13:21 PM
TEMPBOOKING2	TABLE	6/13/2024 1:16:30 PM
TEMPBOOKINGACTIVITIES	TABLE	6/9/2024 11:52:39 PM

User <CURRENT USER>

Oracle ExportSQL InsertsPL/SQL DeveloperLog

☐ Drop tables

☒ Create tables

☐ Truncate tables

☒ Delete records

☒ Disable triggers

☒ Disable foreign key constraints

☒ Include storage

☒ Include privileges

Commit every 100 records (0 = never)

☐ Zip

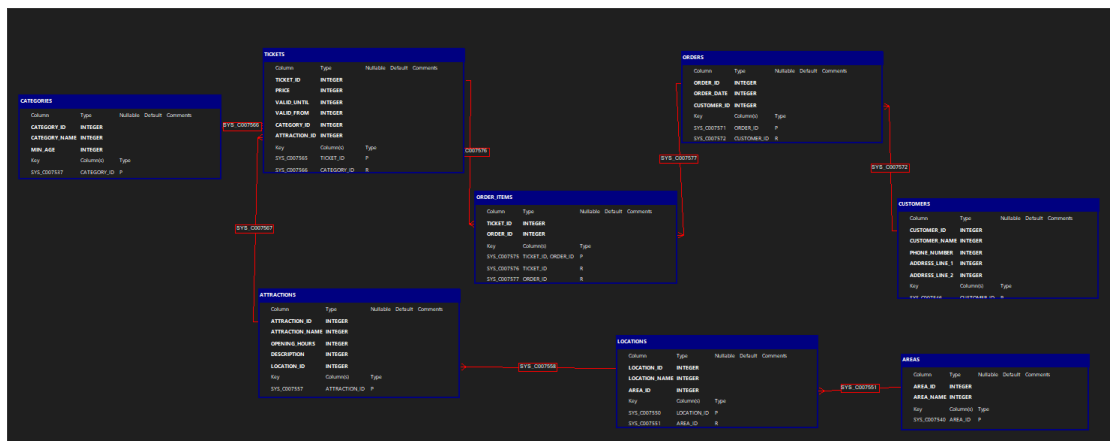
Where clause

Output file C:\Users\linoy\OneDrive\מסלול העבודה\database\שילוב\backup30.6

Export

hotel@XEExporting... Done

## הDSD החדש שקיבלנו בנושא של מכירת כרטיסים לאטרקציות:



## פירוט מסקנות מהDSD ליצירת ERD:

### טבלת Categories:

3 עמודות. מפתח ראשי-category\_id  
קשרים: מקושר לטבלת Tickets

### טבלת Areas:

2 עמודות. מפתח ראשי-area\_id  
קשרים: מקושר לטבלת Locations

### טבלת Customers:

5 עמודות. מפתח ראשי-customer\_id  
קשרים: מקושר לטבלת Orders ע"י קשר Order\_item

### טבלת Locations:

3 עמודות. מפתח ראשי-location\_id. מפתח זר-area\_id  
קשרים: מקושר לטבלת Area ולטבלת Attractions

### טבלת Attractions:

5 עמודות: מפתח ראשי-attraction\_id. מפתח זר-location\_id  
קשרים: מקושר לטבלת Locations ולטבלת Tickets

### טבלת Tickets:

6 עמודות. מפתח ראשי-ticket\_id. מפתחות זרים-attraction\_id , category\_id

קשרים: מקושר לטבלת Attractions ולטבלת Category

### טבלת Orders:

3 עמודות: מפתח ראשי-order\_id, מפתח זר-customer\_id

קשרים: מקושר לטבלת Customers

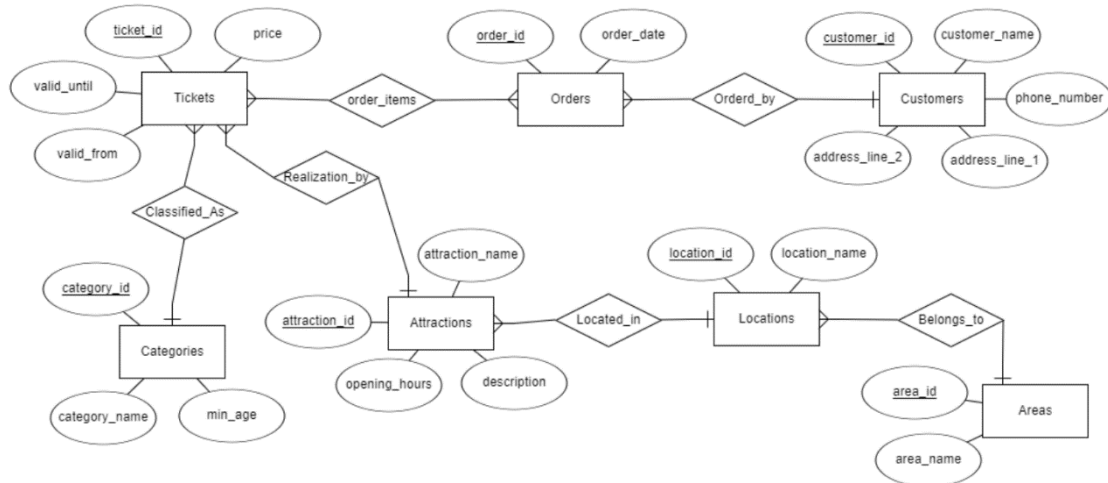
### טבלת order\_items: טבלה של קשר

2 עמודות: מפתח ראשי שמורכב משני מפתחות זרים-ticket\_id, order\_id

קשרים: מקושר לטבלת Orders ולטבלת Tickets

שמות הישויות הן שמות הטבלאות. שמות התכונות הן שמות העמודות בהתאם לכל טבלה.  
מצוין טבלה אחת שמציינת קשר של רבים לרבים, כל שאר הקשרים מציינים קשר של רבים ליחיד.

## ה ERD החדש:



## פקודות עיצוב:

### פקודת עיצוב 1:

מאחר ובשני הפרוייקטים יש טבלה המייצגת לקוחות, מחקנו את הטבלה של הלקוחות מהפרוייקט שקיבלנו והעברנו את כל הקשרים שהיו מחוברים לטבלה זו, אל הטבלה של הלקוחות שלנו, בהתאם לשלבים המוצגים בקוד הבא העברנו את הקשר של טבלת ORDERS מטבלת CUSTOMERS של הפרוייקט שקיבלנו אל טבלת CUSTOMER של הפרוייקט שלנו.

```
--מחיקת כפילות של שני טבלאות המייצגות לקוח--
-- הסרת המפתח הזר הקיים מהטבלה שאותה נמחק
ALTER TABLE Orders DROP CONSTRAINT orders_customer_id_fk;
-- הוספת עמודה חדשה 'c_id'
ALTER TABLE Orders ADD c_id INT;
-- הסרת העמודה הישנה 'customer_id'
ALTER TABLE Orders DROP COLUMN customer_id;
-- Customer-עדכון העמודה החדשה בערכים מ
BEGIN
  FOR rec IN (SELECT order_id FROM Orders) LOOP
    UPDATE Orders
      SET c_id = (SELECT c_id FROM (SELECT c_id FROM Customer ORDER BY DBMS_RANDOM.VALUE) WHERE ROWNUM = 1)
      WHERE order_id = rec.order_id;
  END LOOP;
END;
-- הוספת המפתח הזר עם הקשר הנכון
ALTER TABLE Orders ADD CONSTRAINT orders_customer_id_fk FOREIGN KEY (c_id) REFERENCES Customer(c_id);
select * from orders;
DROP TABLE Customers;
```

## פקודת עיצוב 2:

מאחר ובשני הפרוייקטים יש טבלה המייצגת פעילות/אטרקציה, מחקנו מהפרוייקט שלנו את הטבלה המייצגת פעילויות והעברנו את כל הקשרים של הטבלאות שהיו מחוברות לטבלה זו, אל הטבלה של האטרקציות מהפרוייקט השני בהתאם לפקודות בקוד הבא התאמנו את הטבלאות, WORK, BOOKINGACTIVITIES שיחררנו את הקשר שלהם מטבלת ACTIVITIES לטבלת ATTRACTIONS.

```
--בהתאם Attraction והעברת הקשרים שלה לטבלת Activities מחיקת טבלת--
-- attraction_id ל-a_id שם העמודה
ALTER TABLE Attractions RENAME COLUMN attraction_id TO a_id;
ALTER TABLE Tickets RENAME COLUMN attraction_id TO a_id;

-- שלב 1: הסרת המפתח הזר הקיים
ALTER TABLE WORK DROP CONSTRAINT SYS_C007204;

-- שלב 2: הסרת המפתח הראשי כדי שנוכל למחוק את העמודה
ALTER TABLE WORK DROP CONSTRAINT SYS_C007202;

-- שלב 3: מחיקת העמודה A_ID
ALTER TABLE WORK DROP COLUMN A_ID;

-- מחיקת ערכי הטבלה על מנת שנוכל למלאה מחדש בהתאם--
TRUNCATE TABLE WORK;

-- מחדש A_ID שלב 4: הוספת העמודה
ALTER TABLE WORK ADD A_ID INT NOT NULL;

-- Attractions ו- Employees שורות רנדומליות עם ערכים מטבלאות 20-שלב 7: מילוי הטבלה ב
BEGIN
FOR i IN 1..20 LOOP
INSERT INTO WORK (E_ID, A_ID)
SELECT
(SELECT E_ID FROM (SELECT E_ID FROM EMPLOYEES ORDER BY DBMS_RANDOM.VALUE) WHERE ROWNUM = 1),
(SELECT A_ID FROM (SELECT A_ID FROM ATTRACTIONS ORDER BY DBMS_RANDOM.VALUE) WHERE ROWNUM = 1)
FROM DUAL
WHERE NOT EXISTS (
SELECT 1 FROM WORK
WHERE E_ID = (SELECT E_ID FROM (SELECT E_ID FROM EMPLOYEES ORDER BY DBMS_RANDOM.VALUE) WHERE ROWNUM = 1)
AND A_ID = (SELECT A_ID FROM (SELECT A_ID FROM ATTRACTIONS ORDER BY DBMS_RANDOM.VALUE) WHERE ROWNUM = 1)
);
END LOOP;

-- מחיקת BookingActivities מחיקת כל השורות בטבלה
TRUNCATE TABLE BookingActivities;

-- שלב 1: הסרת המפתח הזר הקיים
ALTER TABLE BookingActivities DROP CONSTRAINT SYS_C007199;

-- שלב 2: הסרת המפתח הראשי (אם יש צורך, דאגי לבדוק אם יש מפתח ראשי קודם שיש להסיר אותו)
ALTER TABLE BookingActivities DROP CONSTRAINT SYS_C007197;

-- (אם צריך לבצע שינוי בעמודה הזו) a_id שלב 3: מחיקת העמודה
ALTER TABLE BookingActivities DROP COLUMN a_id;

-- מחדש a_id שלב 4: הוספת העמודה
ALTER TABLE BookingActivities ADD a_id INT NOT NULL;

-- a_id ו-b_id שלב 5: הוספת המפתח הראשי מחדש על שני העמודות
ALTER TABLE BookingActivities ADD CONSTRAINT SYS_C007197 PRIMARY KEY (b_id, a_id);

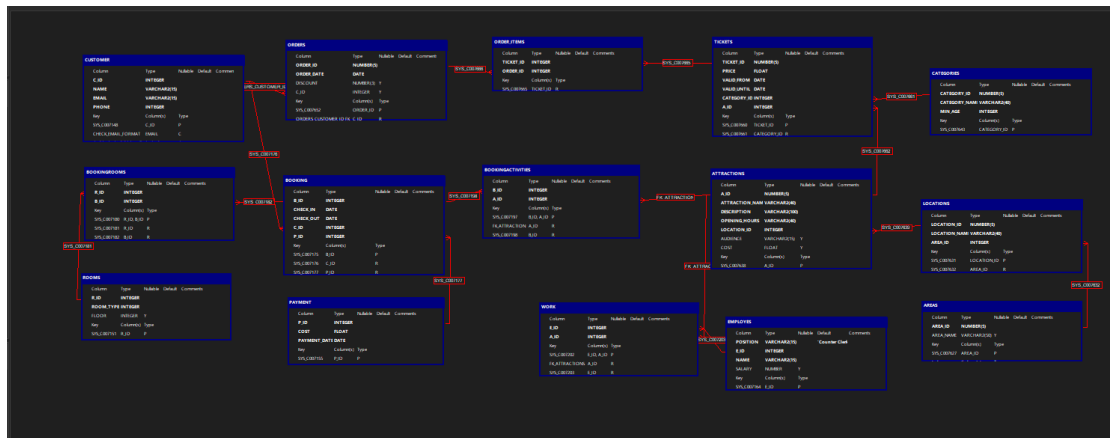
-- Attractions שלב 6: הוספת המפתח הזר מחדש שמקשר לטבלה
ALTER TABLE BookingActivities ADD CONSTRAINT fk_attraction FOREIGN KEY (a_id) REFERENCES Attractions(a_id);

-- Attractions ו- Booking שורות רנדומליות עם ערכים מטבלאות 20-שלב 7: מילוי הטבלה ב
BEGIN
FOR i IN 1..300 LOOP
INSERT INTO BookingActivities (b_id, a_id)
SELECT
(SELECT b_id FROM (SELECT b_id FROM Booking ORDER BY DBMS_RANDOM.VALUE) WHERE ROWNUM = 1),
(SELECT a_id FROM (SELECT a_id FROM Attractions ORDER BY DBMS_RANDOM.VALUE) WHERE ROWNUM = 1)
FROM DUAL
WHERE NOT EXISTS (
SELECT 1 FROM BookingActivities
WHERE b_id = (SELECT b_id FROM (SELECT b_id FROM Booking ORDER BY DBMS_RANDOM.VALUE) WHERE ROWNUM = 1)
AND a_id = (SELECT a_id FROM (SELECT a_id FROM Attractions ORDER BY DBMS_RANDOM.VALUE) WHERE ROWNUM = 1)
);
END LOOP;
END;

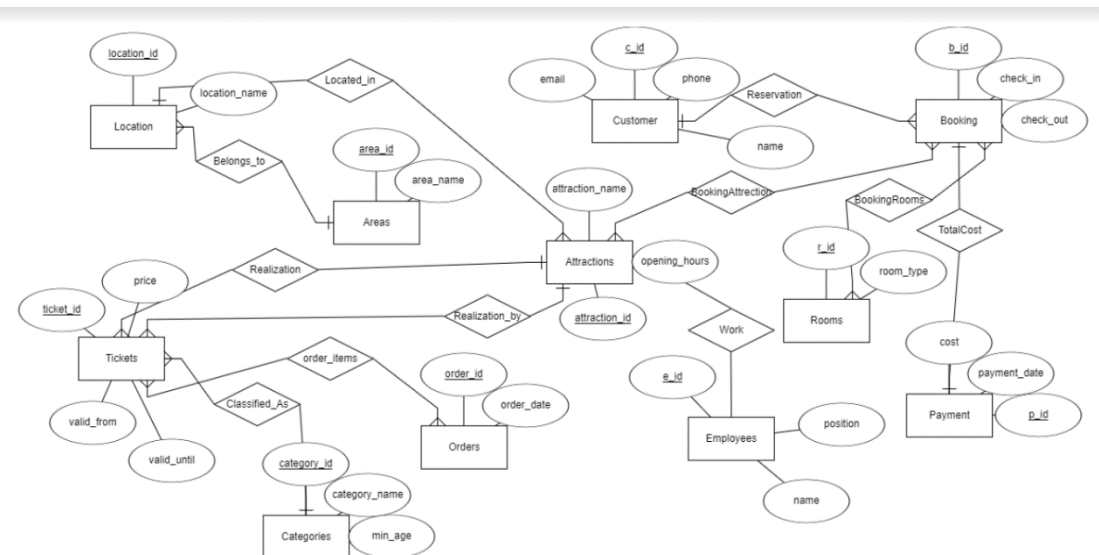
-- Activities מחיקת הטבלה
drop table Activities;

select * from bookingactivities;
select * from work;
select * from attractions;
select * from tickets;
```

## הDSD המשולב:



## הERD המשולב:



## פקודות VIEWS:

### VIEW 1:

יצירת VIEW מנקודת מבט של הפרוייקט שלנו

```
--המידע המוצג כאן ייתן סקירה של ההזמנות, הלקוחות, התשלומים והחדרים.
CREATE VIEW BookingReview AS
SELECT
    b.b_id AS Booking_ID,
    c.name AS Customer_Name,
    c.email AS Customer_Email,
    c.phone AS Customer_Phone,
    p.cost AS Payment_Cost,
    p.payment_date AS Payment_Date,
    COUNT(b.b_id) AS Number_of_Rooms
FROM Booking b
JOIN Customer c ON b.c_id = c.c_id
JOIN Payment p ON b.p_id = p.p_id
JOIN BookingRooms br ON b.b_id = br.b_id
JOIN Rooms r ON br.r_id = r.r_id
GROUP BY b.b_id, c.name, c.email, c.phone, p.cost, p.payment_date;

select * from BookingReview;
```

	BOOKING_ID	CUSTOMER_NAME	CUSTOMER_EMAIL	CUSTOMER_PHONE	PAYMENT_COST	PAYMENT_DATE	NUMBER_OF_ROOMS	
1	65	Bo-Shearer	bos@kellogg.au	548834565	9861	11/5/2023	...	1
2	38	Vince-Sinise	vince.sinise@ca	587445070	6810	3/29/2022	...	1
3	425	Meredith-Vinton	Meredith@com	564622237	8116	11/19/2022	...	1
4	499	Dwight-Bell	dwright.bell@tot	553670508	6100	8/31/2022	...	1
5	408	Debbie-Buscemi	dbuscemi@biorel	517952566	9197	12/17/2023	...	3
6	500	Iann-Bohards	iann.bohards@ca	519672727	6922	11/2/2022	...	2

hotel@XE [10:21:06 PM] 114 rows selected in 0.227 seconds

### שאלתה 1:

```
--שאלתה שמחזירה את שמות הלקוחות והאטרקציות שלהם אלו שהזמינו 4 או יותר חדרים בהזמנה אחת
select br.Customer_Name, a.attraction_name
from BookingReview br JOIN booking b ON br.Booking_ID=b.b_id JOIN bookingactivities ba
ON b.b_id=ba.b_id JOIN attractions a ON ba.a_id=a.a_id
where br.Number_of_Rooms >=4
```

	CUSTOMER_NAME	ATTRACTION_NAME
1	Sophie-Dickinso	Aquarium
2	Sigourney-Hauer	Disneyland
3	Frank-Cole	Disneyland
4	Sophie-Dickinso	Famous Bridge

### שאלתה 2:

```
--שאלתה שמחזירה פרטי לקוח ששכרם ההזמנה שלו עולה על 5000 ומשלם לפני תאריך יציאה מהמלון
select br.Customer_Name, br.Customer_Email, br.Customer_Phone
from BookingReview br JOIN booking b ON br.Booking_ID=b.b_id
where br.Payment_Cost > 5000 and br.payment_date < b.check_out;
```

	CUSTOMER_NAME	CUSTOMER_EMAIL	CUSTOMER_PHONE
1	Liev-Sedaka	liev@totalenter	581094251
2	Bobby-Chaplin	bobby.chaplin@h	525315563
3	Elizabeth-Remar	eremar@swp.com	521931936
4	Mekhi-Palmieri	m.palmieri@apex	533387065
5	Nicky-Bush	nicky.bush@uem.	593206710
6	Praga-Holliday	pragah@flavorx.	537944782
7	Iann-Bohards	iann.bohards@ca	519672727

## VIEW:2

יצירת VIEW מנקודת מבט של הפרוייקט שקיבלנו

```
-- המידע המוצג כאן ייתן סקירה של האטרקציות, הכרטיסים, הקטגוריות וההזמנות.
CREATE VIEW AttractionOverview AS
SELECT
  a.a_id AS Attraction_ID,
  a.attraction_name AS Attraction_Name,
  l.location_name AS Location_Name,
  t.price AS Ticket_Price,
  t.valid_from AS Ticket_Valid_From,
  t.valid_until AS Ticket_Valid_Until,
  c.category_name AS Category_Name
FROM Attractions a
JOIN Locations l ON a.location_id = l.location_id
JOIN Tickets t ON a.a_id = t.a_id
JOIN Categories c ON t.category_id = c.category_id;

select * from AttractionOverview;
```

	Attraction_ID	Attraction_Name	Location_Name	Ticket_Price	Ticket_Valid_From	Ticket_Valid_Until	Category_Name
1	1	Observation Deck	Webster Groves	36	1/10/2024	1/23/2026	Adult
2	1	Observation Deck	Webster Groves	31	1/6/2024	1/13/2026	Child
3	1	Observation Deck	Webster Groves	16	1/22/2024	1/7/2026	Adult
4	1	Observation Deck	Webster Groves	85	1/20/2024	1/16/2026	Teenager
5	1	Observation Deck	Webster Groves	1	1/13/2024	1/25/2026	Child
6	2	Water Park	Archbold	1	1/14/2024	1/24/2026	Teenager
7	3	Observation Deck	San Francisco	52	1/25/2024	1/20/2026	Adult
8	4	Wildlife Sanctuary	Durban	28	1/29/2024	1/10/2026	Child

hotel@XE [10:26:10 PM] 400 rows selected in 0.306 seconds

## שאלתה:1

```
-- שאלתה שמחזירה את מחיר כרטיס הי מוזל בתאריכים של מקום מסוים לקהל יעד מסוים בהתאם להכנסת הנתונים--
select aa.Ticket_Price, aa.Ticket_Valid_From, aa.Ticket_Valid_Until
from AttractionOverview aa
where aa.category_name = &<name = "Category_Name" type="string" list="select category_name from Categories">
and aa.attraction_name = &<name = "attraction_name" type="string" list="select attraction_name from Attractions">
and aa.location_name = &<name = "location_name" type="string" list="select location_name from locations">
and aa.Ticket_Price <= all (select a.Ticket_Price
from AttractionOverview a
where a.category_name = aa.category_name
and a.attraction_name = aa.attraction_name
and a.location_name = aa.location_name);
```

	Ticket_Price	Ticket_Valid_From	Ticket_Valid_Until
1	16	1/22/2024	1/7/2026

Variables

Name	Value
Category_Name	Adult
attraction_name	Observation Deck
location_name	Webster Groves

OK Cancel Clear

## שאלתה:2

```
-- שאלתה שמחזירה את שמות הלקוחות, שמות האטרקציות, וכמה הזמינו מכל אחת מהן --
select a.attraction_name, count(*) AS booked, c.name AS customer_name
from AttractionOverview a JOIN bookingactivities b ON a.Attraction_ID=b.a_id
JOIN booking bk ON b.b_id=bk.b_id JOIN customer c ON bk.c_id=c.c_id
group by a.Attraction_ID, a.attraction_name, c.name;
```

	Attraction_Name	Booked	Customer_Name
1	Public Garden	3	Ming-Na-Warwick
2	Cultural Festival	2	Shelby-Vaughn
3	Botanical Garden	1	Lee-Dourif
4	Observation Deck	2	Meredith-Vinton
5	Aquarium	1	Debbie-Buscemi
6	Art Gallery	1	Roddy-Quinlan
7	Luna Park	1	Teena-Witt
8	Museum	1	Victoria-Bell



## גיבוי 4:

Name	Type	Compiled
AREAS	TABLE	7/18/2024 2:21:44 PM
ATTRACTIONS	TABLE	7/19/2024 1:18:17 PM
BOOKING	TABLE	6/9/2024 11:08:33 PM
BOOKINGACTIVITIES	TABLE	7/19/2024 11:58:02 AM
BOOKINGROOMS	TABLE	5/24/2024 4:20:22 PM
CATEGORIES	TABLE	7/18/2024 2:21:44 PM
CUSTOMER	TABLE	6/9/2024 8:36:13 PM
EMPLOYES	TABLE	6/21/2024 6:00:36 PM
LOCATIONS	TABLE	7/18/2024 2:21:44 PM
ORDERS	TABLE	7/18/2024 7:27:34 PM
ORDER_ITEMS	TABLE	7/18/2024 2:21:44 PM
PAYMENT	TABLE	5/24/2024 4:20:22 PM

User: <CURRENT USER>

Oracle Export SQL Inserts PL/SQL Developer

☐ Drop tables

☒ Create tables

☐ Truncate tables

☒ Delete records

☒ Disable triggers

☒ Disable foreign key constraints

☒ Include storage

☒ Include privileges

Commit every 100 records (0 = never)

☐ Zip

Where clause

Output file: C:\Users\lino\OneDrive\העבודה\שולחן\database\שליב\backup20.7

Export