

2024

# מערכת לניהול הזמינות

מינו פרויקט בסיסי נתוניים ה'תשפ"ד

עמנדב ברנר ויאב סרי

## תוכן עניינים

4.....	תיאור מערכת בית המלון
4.....	תיאור מערכת המידע:
4.....	תיאור הישויות:
5.....	הקשרים במערכת המידע:
6.....	תרשיימי הישויות והקשרים
7.....	יצירת הטבלאות לבסיס הנתונים
7.....	יצירת הטבלה Employee :Employee
8.....	יצירת הטבלה ReservationAgent :ReservationAgent
9.....	יצירת הטבלה Receptionist :Receptionist
10.....	יצירת הטבלה Room :Room
11.....	יצירת הטבלה Guest :Guest
13.....	יצירת הטבלה Booking :Booking
14.....	יצירת הטבלה Request :Request
15.....	יצירת הטבלה Inform :Inform
17.....	סיכום יצירה, הכנסה, ומחיקה..
17.....	סיכום יצירת הטבלאות:
18.....	סיכום הכנסת ערכים לטבלאות:
19.....	מחיקת טבלאות:
20.....	הציגת הטבלאות:
21.....	אבלוס בסיס הנתונים
21.....	אבלוס הטבלה Guest :Guest
21.....	אבלוס הטבלה Room :Room
22.....	אבלוס הטבלה Employee :Employee
23.....	אבלוס הטבלה ReservationAgent :ReservationAgent
23.....	אבלוס הטבלה Receptionist :Receptionist
24.....	אבלוס הטבלה Booking :Booking
24.....	אבלוס הטבלה Request :Request
24.....	אבלוס הטבלה Inform :Inform
26.....	גיבוי ו恢復 בסיס הנתונים .....
26.....	גיבוי בסיס הנתונים: .....

יבוא בסיס נתונים:.....	26
שאילות על בסיס הנתונים.....	29
שאילתת select:.....	29
שאילתת 1:.....	29
שאילתת 2:.....	30
שאילתת 3:.....	30
שאילתת 4:.....	31
שאילתות delete:.....	32
שאילתת 1:.....	32
שאילתת 2:.....	33
שאילתות update:.....	35
שאילתת 1:.....	35
שאילתת 2:.....	36
שאילתות עם פרמטרים:.....	38
שאילתת 1:.....	38
שאילתת 2:.....	39
שאילתת 3:.....	41
שאילתת 4:.....	42
הוספת אילוצים לdatableות:.....	44
אילוץ ראשון:.....	44
אילוץ שני:.....	44
אילוץ שלישי:.....	45
תכנות ב- PL/SQL.....	46
פונקציות:.....	46
פונקציה ראשונה:.....	46
פונקציה שנייה:.....	47
פרוצדורות:.....	49
פרוצדורה ראשונה:.....	49
פרוצדורה שנייה:.....	51
תוכניות הראשונות:.....	54
תוכנית ראשונה:.....	54
תוכנית שנייה:.....	55

57.....	אינטגרציה עם אגף ניהול עובדים .....
57.....	הנדסה לאחר של בסיס הנתונים של אגף ניהול עובדים: .....
60.....	אינטגרציה ברמת העיצוב: .....
60.....	תיאור כללי של השינויים: .....
60.....	תרשים ERD משולב של אגף ניהול עובדים ואגף ניהול הזמן: .....
61.....	תרשים DSD משולב של אגף ניהול עובדים ואגף ניהול הזמן: .....
61.....	אינטגרציה ברמת הנתונים: .....
61.....	תיאור מפורט של הדרך לאינטגרציה מלאה: .....
62.....	הקוד שמבצע את האינטגרציה בין האגפים: .....
69.....	מבטים .....
69.....	הגדרת ה- Views: .....
69.....	View ראשון: .....
70.....	View שני: .....
71.....	שאילות על ה- Views: .....
71.....	שאילתת ה- View הראשון: .....
72.....	שאילתת ה- View השני: .....

# תיאור מערכת בית המלון

## תיאור מערכת המידע:

אנחנו יוצרים מאגר נתונים עבור בית המלון הידוע "AirPods Plus Plus" בברית הניה. בית מלון, כמו בית מלון, מורכב מחלקות שונות ומשונות שתפקיד המשותף הוא לדאוג לרוחות האורחים. בחלוקת ניהול הזמן הסתברו עד לא זמן עם טפסים ידניים ונΚראנו לדגל כדי להציג את המצב!

תפקידה של מחלקת הזמן הוא לטפל בכל מה שקשר להזמנות חדרים שבוצעות על ידי אורחים המלון. באשר אורח מתקשר למילון לבצע הזמנה הוא מטופל על ידי אחד העובדים מצוות הדלקאים של המלון. אם צרי, הדלקאי אותו הוא מדובר יכול להפנות אותו לאחד מסוכני הזמן של בית המלון בצד שידע אותו בכל מה שהוא צריך לגבי הזמן חדר.

## תיאור היחסיות:

- **אורח (Guest):** טבלת האורחים תכילה את כל האורחים שנקלטו במערכת המלון משנת 1970

עד היום (כולל אלו העתידיים להതארה).

guest\_id – מספר תעודה זהות.

first\_name – שם פרטי.

last\_name – שם משפחה.

phone – מספר טלפון ליצירת קשר.

date\_of\_birth – תאריך לידה.

address – כתובת מגורים.

- **חדר (Room):** טבלת חדרי השינה תכילה את כל חדרי השינה שבמלון.

room\_number – מספר החדר.

beds – מספר מיטות בחדר.

balcony – מרפסת (יש / אונ).

price – מחיר ללילה.

- **עובד (Employee):** טבלת העובדים תכילה את כל עובדי בית המלון לסוגם.

em\_id – מספר העובד.

first\_name – שם פרטי.

last\_name – שם משפחה.

salary – משכורת חודשית.

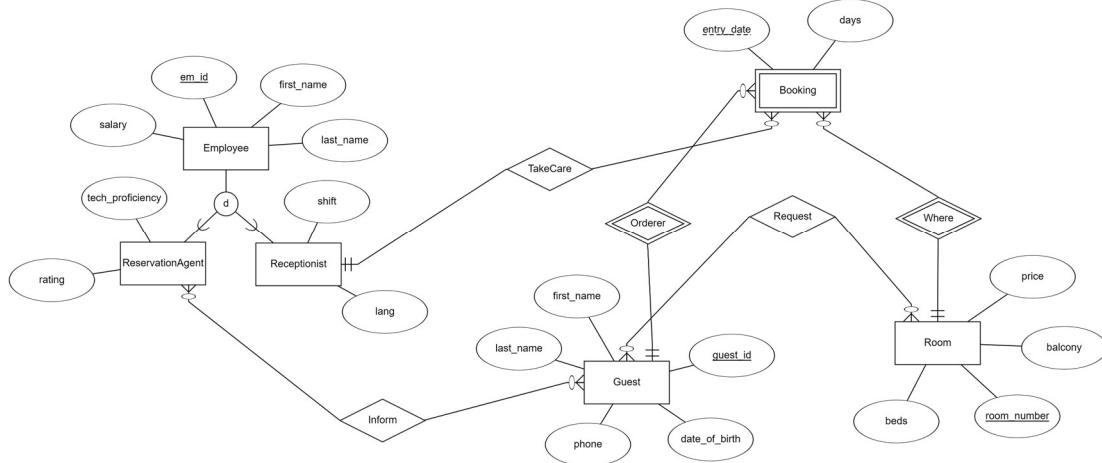
- **סוכן (ReservationAgent):** טבלת הסוכנים מכיל את כל הסוכנים של בית המלון.
  - em\_id – מספר העובד.
  - rating – דירוג שקיבל הסוכן מהלקוחות.
  - tech\_proficiency – התמחות טכנית (רמה גבוהה, ממוצעת, נמוכה).
- **מארח (Receptionist):** טבלת המארחים מכיל את כל המארחים של בית המלון.
  - em\_id – מספר העובד.
  - lang – שפת אם לתקשורת עם לקוחות.
  - shift – משמרת עבודה (בוקר / ערבית / לילה).
- **הזמנה (Booking):** טבלת ההזמנות מכיל את כל ההזמנות של האורחים בבית המלון.
  - guest\_id – תעודה זהות המזמין.
  - room\_number – מספר החדר.
  - entry\_date – תאריך כניסה.
  - days – מספר ימי שהוות במלון.

#### הקשרים במערכת המידע:

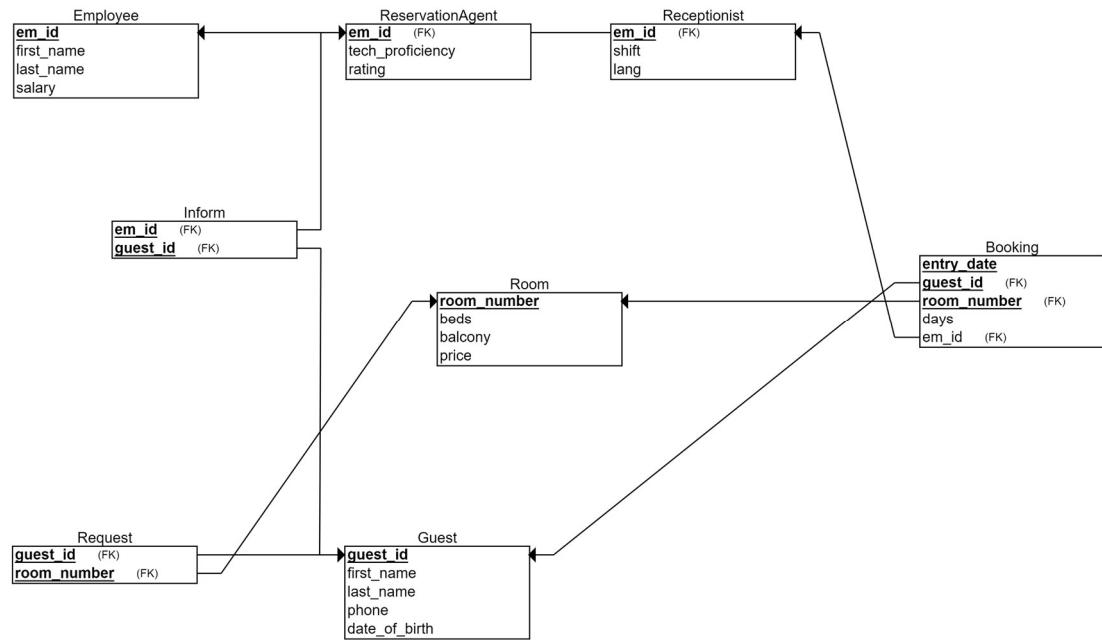
- **קשר Inform:** לא כל האורחים שורצים להזמין חדרים במלון בקיאים בסוד העניינים. יש כאלו שזו פעם ראשונה שהם מזמינים עצמם בחדר/ים בבית מלון. לצורך כך המלון מעסיק סוכנים שתפקידם לסייע לאוטם אנשים לבצע את ההזמנה עליה הם חולמים.
- **קשר TakeCare:** כאשר מתאפשרת הזמנה במערכת, היא מועברת לטיפול אחד מהמאורחים של בית המלון. כל הזמנה מטופלת על ידי מארח אחד בלבד לאורך כל חייה.
- **קשר Request:** במהלך שהותם של האורחים בבית המלון הם יכולים להיתקל בחדרים שדורשים טיפול או תיקון (גם אם זה לא החדר שם הזמין). טיפול יכול להיות גם לבקש מגבת נוספת או כל דבר אחר. במקרה זה הם ימלאו טופס דיווח על תקלת שישלח לאגף התחזקה של בית המלון. כל אורח יוכל לדוח על כל אחד מהחדרים, ולהילופין – רבים לרבים.

# תרשיימי הישויות והקשרים

התרשים הראשון הוא תרשים ERD שמתאר לנו את התכונות של כל ישות ואת הקשרים בין ישות אחת לחברתה.



התרשים השני של המערכת הוא תרשים DSD שמציג את הקשרים בין התכונות של הישויות. בתרשים זה נובל לראות איזו ישות תליה באיזו ישות מבחינת המפתח הזר שלה וכו'.



את שני התרשיימים יצרנו בעזרת האתר <https://erdplus.com>

# יצירת הטבלאות לבסיס הנתונים

ביצירת הטבלאות נשים לב שיש טבלאות שלא ניתן ליצור לפני שייצרנו טבלה קודמת, למשל את הטבלה Receptionist לא יוכל ליצור לפני שייצרנו את הטבלה Employee, כיון שהתכונה em\_id ב-Table Receptionist היא מפתחZR.

הסדר שבו נבחר ליצור את הטבלאות הוא (משמאל לימין):

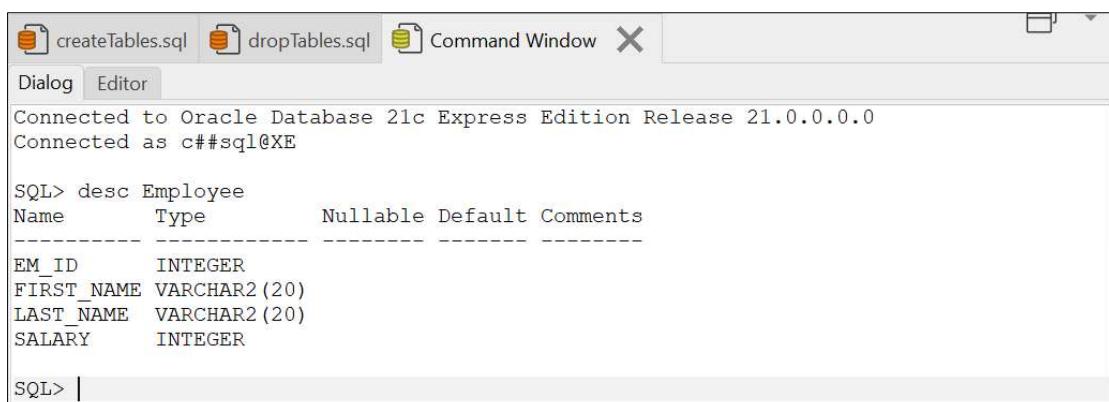
Employee, ReservationAgent, Receptionist, Room, Guest, Booking, Request, Inform

## יצירת הטבלה Employee:

קוד ה-SQL שיוצר לנו את הטבלה הוא:

```
CREATE TABLE Employee
(
    em_id INT NOT NULL,
    first_name VARCHAR2(20) NOT NULL,
    last_name VARCHAR2(20) NOT NULL,
    salary INT NOT NULL,
    PRIMARY KEY (em_id)
);
```

בדוק שakan הטבלה נוצרה בסיס הנתונים:



The screenshot shows the Oracle Database 21c Express Edition Release 21.0.0.0.0 connected as c##sql@XE. In the Command Window, the command `desc Employee` is run, displaying the table structure:

Name	Type	Nullable	Default	Comments
EM_ID	INTEGER			
FIRST_NAME	VARCHAR2(20)			
LAST_NAME	VARCHAR2(20)			
SALARY	INTEGER			

וכניס לטבלה מספר ערכים על מנת לבדוק את תקינותה של הסכמה:

```
INSERT INTO Employee (em_id, first_name, last_name, salary)
VALUES (1, 'John', 'Doe', 50000);
INSERT INTO Employee (em_id, first_name, last_name, salary)
VALUES (2, 'Jane', 'Smith', 55000);
INSERT INTO Employee (em_id, first_name, last_name, salary)
VALUES (3, 'Alice', 'Johnson', 60000);
```

בדוק שakan הערכים הוכנסו לטבלה כמו שצריך:

The screenshot shows the Oracle SQL Developer interface. At the top, there are four tabs: 'createTables.sql', 'dropTables.sql', 'insertTables.sql', and 'Query data C##SQL.EMPLOYEE@XE'. The 'Query data C##SQL.EMPLOYEE@XE' tab is active. Below it, the SQL tab contains the query: 'select \* from EMPLOYEE t'. The results pane displays the following data:

	EM_ID	FIRST_NAME	LAST_NAME	SALARY
▶	1	John	Doe	50000
	2	Jane	Smith	55000
	3	Alice	Johnson	60000

### : יצירת הטבלה ReservationAgent

קוד ה-SQL שיצרת לנו את הטבלה הוא:

```
CREATE TABLE ReservationAgent
(
    em_id INT NOT NULL,
    tech_proficiency VARCHAR2(15) NOT NULL,
    rating INT NOT NULL,
    PRIMARY KEY (em_id),
    FOREIGN KEY (em_id) REFERENCES Employee(em_id)
);
```

נבדוק שacky הטבלה נוצרה מבוסיס הנתונים:

The screenshot shows the Oracle SQL Developer Command Window. It displays the following output:

```
Connected to Oracle Database 21c Express Edition Release 21.0.0.0.0
Connected as c##sql@XE

SQL> desc ReservationAgent
Name          Type      Nullable Default Comments
-----        -----      -----   -----
EM_ID         INTEGER
TECH_PROFICIENCY VARCHAR2(15)
RATING        INTEGER

SQL> |
```

נכнес לatable מספר ערכים על מנת לבדוק את תקינותה של הסכמה:

```
INSERT INTO ReservationAgent (em_id, tech_proficiency, rating)
VALUES (1, 'Expert', 5);
INSERT INTO ReservationAgent (em_id, tech_proficiency, rating)
```

```
VALUES (2, 'Intermediate', 4);
```

נבדוק שacky הערכים הוכנסו לטבלה כמו שצריך:

The screenshot shows the SQL tab of the SSMS interface. A query window displays the following SQL command:

```
select * from RESERVATIONAGENT t
```

Below the query window, the results pane shows the data from the `RESERVATIONAGENT` table:

	EM_ID	TECH_PROFICIENCY	RATING
▶	1	Expert	5
	2	Intermediate	4

### יצירת הטבלה Receptionist

קוד ה-SQL שיוצר לנו את הטבלה הוא:

```
CREATE TABLE Receptionist
(
    em_id INT NOT NULL,
    shift VARCHAR2(10) NOT NULL,
    lang VARCHAR2(30) NOT NULL,
    PRIMARY KEY (em_id),
    FOREIGN KEY (em_id) REFERENCES Employee(em_id)
);
```

נבדוק שacky הטבלה נוצרה מבוסיס הנתונים:

The screenshot shows the Oracle SQL\*Plus environment. The command window displays the following output:

```
Connected to Oracle Database 21c Express Edition Release 21.0.0.0.0
Connected as c##sql@XE

SQL> desc Receptionist
Name      Type          Nullable Default Comments
-----  -----
EM_ID     INTEGER
SHIFT     VARCHAR2(10)
LANG      VARCHAR2(30)

SQL> |
```

נכнес לטבלה מספר ערכים על מנת לבדוק את תקינותה של הסכמה:

```

INSERT INTO Receptionist (em_id, shift, lang)
VALUES (3, 'Night', 'English');
INSERT INTO Receptionist (em_id, shift, lang)
VALUES (2, 'Day', 'French');

```

בדוק שancock הערבים הוכנסו לטבלה כמו שצריך:

The screenshot shows the SSMS interface with the Query data C##SQL.RECEPTIONIST window active. The SQL tab contains the query: `select * from RECEPTIONIST t`. Below the query results, the RECEPTIONIST table is displayed in a grid format. The table structure is as follows:

	EM_ID	SHIFT	LANG
▶	1	3	Night
	2	2	Day
			French

### יצירת הטבלה Room:

קוד ה-SQL שיוצר לנו את הטבלה הוא:

```

CREATE TABLE Room
(
    room_number INT NOT NULL,
    beds INT NOT NULL,
    balcony VARCHAR2(4) NOT NULL,
    price INT NOT NULL,
    PRIMARY KEY (room_number)
);

```

בדוק שancock הטבלה נוצרה מבוסיס הנתונים:

```

Connected to Oracle Database 21c Express Edition Release 21.0.0.0.0
Connected as c##sql@XE

SQL> desc Room
Name          Type      Nullable Default Comments
-----
ROOM_NUMBER   INTEGER   NO        NULL    ROOM NUMBER
BEDS          INTEGER   NO        NULL    BEDS
BALCONY       VARCHAR2(4) NO        NULL    BALCONY
PRICE         INTEGER   NO        NULL    PRICE

```

נכнес לatable מספר ערכים על מנת לבדוק את תקינותה של הסכמה:

```

INSERT INTO Room (room_number, beds, balcony, price)
VALUES (101, 2, 'Yes', 100);
INSERT INTO Room (room_number, beds, balcony, price)
VALUES (102, 1, 'No', 80);
INSERT INTO Room (room_number, beds, balcony, price)
VALUES (103, 3, 'Yes', 150);

```

בדוק שacky הערכים הוכנסו לatable כמו שצריך:

	ROOM_NUMBER	BEDS	BALCONY	PRICE
1	101	2	Yes	100
2	102	1	No	80
3	103	3	Yes	150

### יצירת הטבלה Guest

קוד ה-SQL שיוצר לנו את הטבלה הוא:

```

CREATE TABLE Guest
(
  guest_id INT NOT NULL,
  first_name VARCHAR2(20) NOT NULL,

```

```

    last_name VARCHAR2(20) NOT NULL,
    phone VARCHAR2(11) NOT NULL,
    date_of_birth DATE NOT NULL,
    PRIMARY KEY (guest_id)
);

```

נבדוק שacky הטבלה נוצרה מבוסיס הנתונים:

```

Connected to Oracle Database 21c Express Edition Release 21.0.0.0.0
Connected as c##sql@XE

SQL> desc Guest
Name          Type        Nullable Default Comments
-----        -----      -----
GUEST_ID      INTEGER
FIRST_NAME    VARCHAR2(20)
LAST_NAME     VARCHAR2(20)
PHONE         VARCHAR2(11)
DATE_OF_BIRTH DATE

SQL>

```

נכнес לטבלה מספר ערכים על מנת לבדוק את תקינותה של הסכמה:

```

INSERT INTO Guest (guest_id, first_name, last_name, phone,
date_of_birth)
VALUES (1, 'Michael', 'Brown', 1234567890, DATE '1985-07-15');
INSERT INTO Guest (guest_id, first_name, last_name, phone,
date_of_birth)
VALUES (2, 'Sarah', 'Davis', 9876543210, DATE '1990-12-05');

```

נבדוק שacky הערכים הוכנסו לטבלה כמו שצירוי:

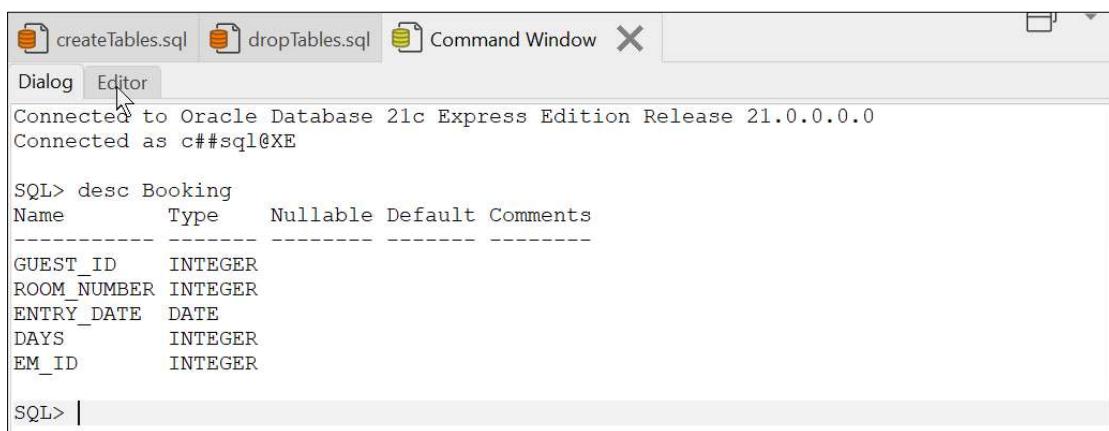
	GUEST_ID	FIRST_NAME	LAST_NAME	PHONE	DATE_OF_BIRTH
▶	1	Michael	... Brown	... 1234567890	15/07/1985 ...
	2	Sarah	... Davis	... 9876543210	05/12/1990 ...

## יצירת הטבלה :Booking

קוד ה-SQL שיוצר לנו את הטבלה הוא:

```
CREATE TABLE Booking
(
    guest_id INT NOT NULL,
    room_number INT NOT NULL,
    entry_date DATE NOT NULL,
    days INT NOT NULL,
    em_id INT NOT NULL,
    PRIMARY KEY (entry_date, guest_id, room_number),
    FOREIGN KEY (guest_id) REFERENCES Guest(guest_id),
    FOREIGN KEY (room_number) REFERENCES Room(room_number),
    FOREIGN KEY (em_id) REFERENCES Receptionist(em_id)
);
```

נבדוק שacky הטבלה נוצרה מבוסיס הנתונים:



The screenshot shows the Oracle SQL Developer interface. The title bar says "createTables.sql" and "dropTables.sql" are open. The "Command Window" tab is active. The window displays the following text:

```
Connected to Oracle Database 21c Express Edition Release 21.0.0.0.0
Connected as c##sql@XE

SQL> desc Booking
Name          Type      Nullable Default Comments
-----        -----
GUEST_ID      INTEGER   NO        NO
ROOM_NUMBER   INTEGER   NO        NO
ENTRY_DATE    DATE     NO        NO
DAYS          INTEGER   NO        NO
EM_ID         INTEGER   NO        NO

SQL> |
```

נכnis לatable מספר ערכים על מנת לבדוק את תקינותה של הסכמה:

```
INSERT INTO Booking (guest_id, room_number, entry_date, days, em_id)
VALUES (1, 101, DATE '2024-07-01', 3, 3);
INSERT INTO Booking (guest_id, room_number, entry_date, days, em_id)
VALUES (2, 102, DATE '2024-07-02', 2, 2);
```

נבדוק שacky הערכים הוכנסו לatable כמו שצרי:

The screenshot shows the SQL tab of the SSMS interface. At the top, there are four tabs: 'createTables.sql', 'dropTables.sql', 'insertTables.sql', and 'Query data C##SQL.BOOKING@XE'. Below the tabs, the SQL command `select \* from BOOKING t` is entered. The results pane displays the following data:

	GUEST_ID	ROOM_NUMBER	ENTRY_DATE	END_DATE	DAYS	EM_ID
▶	1	1	101	01/07/2024	...	3
	2	2	102	02/07/2024	...	2

### :Request הטבלה

קוד ה-SQL שיצרת לנו את הטבלה הוא:

```
CREATE TABLE Request
(
    guest_id INT NOT NULL,
    room_number INT NOT NULL,
    PRIMARY KEY (guest_id, room_number),
    FOREIGN KEY (guest_id) REFERENCES Guest(guest_id),
    FOREIGN KEY (room_number) REFERENCES Room(room_number)
);
```

נבדוק שacky הטבלה נוצרה מבוסיס הנתונים:

The screenshot shows the Command Window of Oracle SQL Developer. It displays the following output:

```
Connected to Oracle Database 21c Express Edition Release 21.0.0.0.0
Connected as c##sql@XE

SQL> desc Request
Name          Type   Nullable Default Comments
-----        -----  -----
GUEST_ID      INTEGER
ROOM_NUMBER   INTEGER

SQL>
```

וכניס לatable מספר ערכים על מנת לבדוק את תקינותה של הסכמה:

```
INSERT INTO Request (guest_id, room_number)
VALUES (1, 101);
INSERT INTO Request (guest_id, room_number)
VALUES (2, 102);
```

נבדוק שacky הערכים הוכנסו לatable כמו שצריך:

The screenshot shows the Oracle SQL Developer interface. At the top, there are four tabs: 'createTables.sql', 'dropTables.sql', 'insertTables.sql', and 'Query data C##SQL.REQUEST@XE'. Below these tabs, there are three tabs: 'SQL', 'Output', and 'Statistics', with 'SQL' selected. In the main SQL editor area, the following query is written:  
select \* from REQUEST t

Below the editor is a grid-based data viewer. It has two columns: 'GUEST\_ID' and 'ROOM\_NUMBER'. There are two rows of data:

	GUEST_ID	ROOM_NUMBER
▶	1	101
	2	102

### יצירת הטבלה Inform:

קוד ה-SQL שיוצר לנו את הטבלה הוא:

```
CREATE TABLE Inform
(
    em_id INT NOT NULL,
    guest_id INT NOT NULL,
    PRIMARY KEY (em_id, guest_id),
    FOREIGN KEY (em_id) REFERENCES ReservationAgent(em_id),
    FOREIGN KEY (guest_id) REFERENCES Guest(guest_id)
);
```

נבדוק שacky הטבלה נוצרה מבוסיס הנתונים:

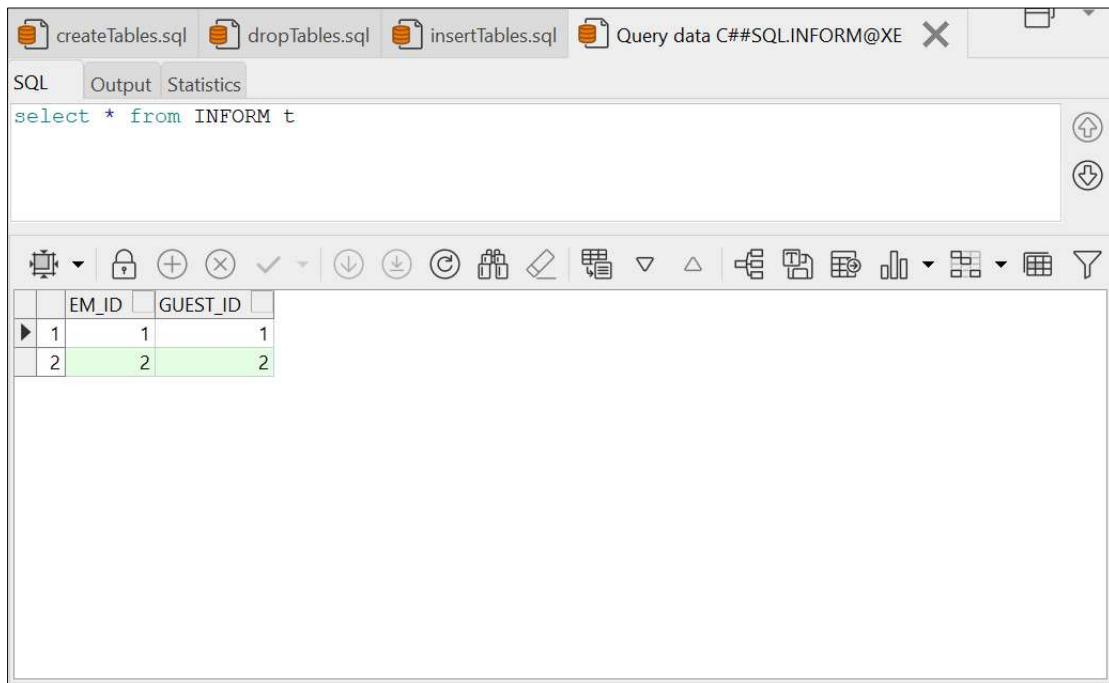
The screenshot shows the Oracle SQL Developer Command Window. At the top, there are three tabs: 'createTables.sql', 'dropTables.sql', and 'Command Window', with 'Command Window' selected. The window displays the following text:  
Connected to Oracle Database 21c Express Edition Release 21.0.0.0.0  
Connected as c##sql@XE  
SQL> desc Inform  
Name Type Nullable Default Comments  
----- ----- -----  
EM\_ID INTEGER  
GUEST\_ID INTEGER  
SQL> |

נכнес לatable מספר ערכים על מנת לבדוק את תקינותה של הסכמה:

```
INSERT INTO Inform (em_id, guest_id)
```

```
VALUES (1, 1);
INSERT INTO Inform (em_id, guest_id)
VALUES (2, 2);
```

נבדוק שאכן הערבים הוכנסו לatable כמו שצראר:



The screenshot shows the SQL tab of the SSMS interface. The query window contains the command: `select * from INFORM t`. Below the query window is a toolbar with various icons for managing tables, columns, rows, and data. Underneath the toolbar is a preview pane displaying the data from the INFORM table. The table has two columns: EM\_ID and GUEST\_ID. There are two rows of data: the first row has EM\_ID 1 and GUEST\_ID 1, and the second row has EM\_ID 2 and GUEST\_ID 2. The second row is highlighted with a green background.

	EM_ID	GUEST_ID
▶	1	1
	2	2

# סיכום יצירה, הכנסה, ומחיקה

## סיכום יצירת הטבלאות:

נאחד את כל קטעי הקוד שכתבנו ליצירת הטבלאות לקובץ אחד בשם `.createTables.sql`

```
-- Table to store employee details
CREATE TABLE Employee
(
    em_id INT NOT NULL,
    first_name VARCHAR2(20) NOT NULL,
    last_name VARCHAR2(20) NOT NULL,
    salary INT NOT NULL,
    PRIMARY KEY (em_id)
);

-- Table to store reservation agent details, extending Employee
CREATE TABLE ReservationAgent
(
    em_id INT NOT NULL,
    tech_proficiency VARCHAR2(15) NOT NULL,
    rating INT NOT NULL,
    PRIMARY KEY (em_id),
    FOREIGN KEY (em_id) REFERENCES Employee(em_id)
);

-- Table to store receptionist details, extending Employee
CREATE TABLE Receptionist
(
    em_id INT NOT NULL,
    shift VARCHAR2(10) NOT NULL,
    lang VARCHAR2(30) NOT NULL,
    PRIMARY KEY (em_id),
    FOREIGN KEY (em_id) REFERENCES Employee(em_id)
);

-- Table to store room details
CREATE TABLE Room
(
    room_number INT NOT NULL,
    beds INT NOT NULL,
    balcony VARCHAR2(4) NOT NULL,
    price INT NOT NULL,
    PRIMARY KEY (room_number)
);

-- Table to store guest details
CREATE TABLE Guest
(
    guest_id INT NOT NULL,
    first_name VARCHAR2(20) NOT NULL,
    last_name VARCHAR2(20) NOT NULL,
    phone VARCHAR2(11) NOT NULL,
    date_of_birth DATE NOT NULL,
    PRIMARY KEY (guest_id)
);

-- Table to store booking details
```

```

CREATE TABLE Booking
(
    guest_id INT NOT NULL,
    room_number INT NOT NULL,
    entry_date DATE NOT NULL,
    days INT NOT NULL,
    em_id INT NOT NULL,
    PRIMARY KEY (entry_date, guest_id, room_number),
    FOREIGN KEY (guest_id) REFERENCES Guest(guest_id),
    FOREIGN KEY (room_number) REFERENCES Room(room_number),
    FOREIGN KEY (em_id) REFERENCES Receptionist(em_id)
);

-- Table to store requests made by guests
CREATE TABLE Request
(
    guest_id INT NOT NULL,
    room_number INT NOT NULL,
    PRIMARY KEY (guest_id, room_number),
    FOREIGN KEY (guest_id) REFERENCES Guest(guest_id),
    FOREIGN KEY (room_number) REFERENCES Room(room_number)
);

-- Table to store information about interactions between reservation
agents and guests
CREATE TABLE Inform
(
    em_id INT NOT NULL,
    guest_id INT NOT NULL,
    PRIMARY KEY (em_id, guest_id),
    FOREIGN KEY (em_id) REFERENCES ReservationAgent(em_id),
    FOREIGN KEY (guest_id) REFERENCES Guest(guest_id)
);

```

### סיכום הכנסת ערכים לטבלאות:

נאחד את כל קטעי הקוד שכתבנו להכנסת ערכים לטבלאות לקובץ אחד בשם `insertTables.sql`

```

-- Inserting into Employee table
INSERT INTO Employee (em_id, first_name, last_name, salary)
VALUES (1, 'John', 'Doe', 50000);
INSERT INTO Employee (em_id, first_name, last_name, salary)
VALUES (2, 'Jane', 'Smith', 55000);
INSERT INTO Employee (em_id, first_name, last_name, salary)
VALUES (3, 'Alice', 'Johnson', 60000);

-- Inserting into ReservationAgent table
INSERT INTO ReservationAgent (em_id, tech_proficiency, rating)
VALUES (1, 'Expert', 5);
INSERT INTO ReservationAgent (em_id, tech_proficiency, rating)
VALUES (2, 'Intermediate', 4);

-- Inserting into Receptionist table
INSERT INTO Receptionist (em_id, shift, lang)
VALUES (3, 'Night', 'English');
INSERT INTO Receptionist (em_id, shift, lang)
VALUES (2, 'Day', 'French');

-- Inserting into Room table
INSERT INTO Room (room_number, beds, balcony, price)

```

```

VALUES (101, 2, 'Yes', 100);
INSERT INTO Room (room_number, beds, balcony, price)
VALUES (102, 1, 'No', 80);
INSERT INTO Room (room_number, beds, balcony, price)
VALUES (103, 3, 'Yes', 150);

-- Inserting into Guest table
INSERT INTO Guest (guest_id, first_name, last_name, phone,
date_of_birth)
VALUES (1, 'Michael', 'Brown', 1234567890, DATE '1985-07-15');
INSERT INTO Guest (guest_id, first_name, last_name, phone,
date_of_birth)
VALUES (2, 'Sarah', 'Davis', 9876543210, DATE '1990-12-05');

-- Inserting into Booking table
INSERT INTO Booking (guest_id, room_number, entry_date, days, em_id)
VALUES (1, 101, DATE '2024-07-01', 3, 3);
INSERT INTO Booking (guest_id, room_number, entry_date, days, em_id)
VALUES (2, 102, DATE '2024-07-02', 2, 2);

-- Inserting into Request table
INSERT INTO Request (guest_id, room_number)
VALUES (1, 101);
INSERT INTO Request (guest_id, room_number)
VALUES (2, 102);

-- Inserting into Inform table
INSERT INTO Inform (em_id, guest_id)
VALUES (1, 1);
INSERT INTO Inform (em_id, guest_id)
VALUES (2, 2);

```

### מחיקת טבלאות:

כפי שביצירת הטבלאות הינו צריכים להקפיד מאוד על הסדר הנכון, כך גם במחיקת הטבלאות علينا להקפיד על הסדר מאותה הסיבה. לא נוכל למחוק טבלה שאחת התכונות שלה היא מפתחZR בטבלה אחרת... הסדר בו נמחק את הטבלאות יהיה בסדר הפוך לסדר אפשרי ליצירתן.

```

-- Drop Inform table
DROP TABLE Inform;

-- Drop Request table
DROP TABLE Request;

-- Drop Booking table
DROP TABLE Booking;

-- Drop Guest table
DROP TABLE Guest;

-- Drop Room table
DROP TABLE Room;

-- Drop Receptionist table
DROP TABLE Receptionist;

-- Drop ReservationAgent table
DROP TABLE ReservationAgent;

```

```
-- Drop Employee table  
DROP TABLE Employee;
```

### הציגת הטבלאות:

נרצה לבתוב קוד SQL שיציג לנו את כל אחת מהטבלאות שיצרנו ואכלסנו (עוד לא ממש אכלסנו, רק הכנסנו מעט מאוד ערכים לבדיקה).

```
-- Select all from Employee table  
SELECT * FROM Employee;  
  
-- Select all from ReservationAgent table  
SELECT * FROM ReservationAgent;  
  
-- Select all from Receptionist table  
SELECT * FROM Receptionist;  
  
-- Select all from Room table  
SELECT * FROM Room;  
  
-- Select all from Guest table  
SELECT * FROM Guest;  
  
-- Select all from Booking table  
SELECT * FROM Booking;  
  
-- Select all from Request table  
SELECT * FROM Request;  
  
-- Select all from Inform table  
SELECT * FROM Inform;
```

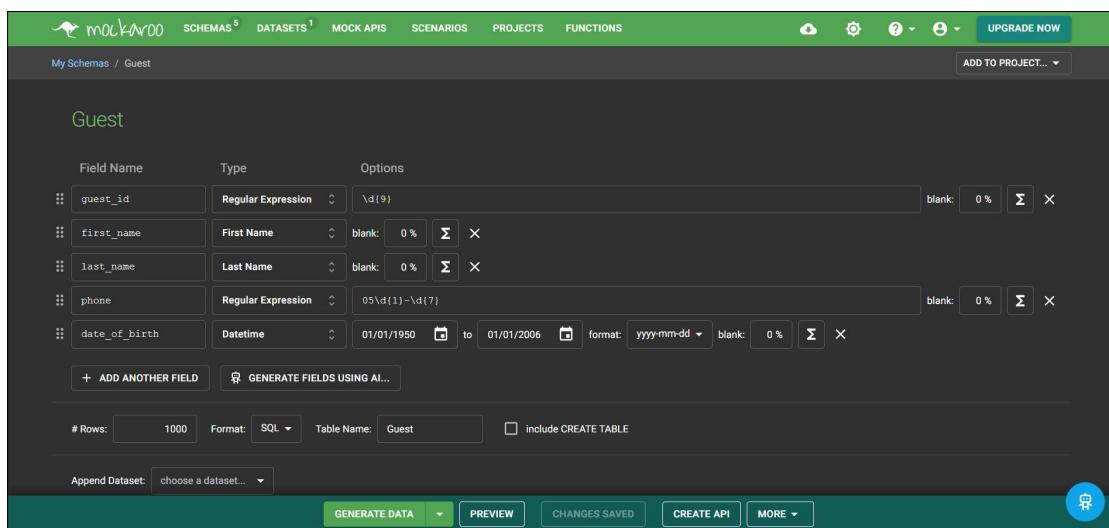
# אבלוס בסיס הנתונים

נשתמש בשלוש דרכים כדי לג'נרט (לחולל בעברית) נתונים לבסיס המידע שלנו:

- הכללי Data Generator שМОבנה בסביבת העבודה של PL/SQL.
- קבצי SQL עם פקודות ליצירת רשומות שניצור באמצעות האתר [Mockaroo](#).
- קבצי Excel עם טבלאות נתונים אותם ניתןนำไป לבסיס הנתונים שלנו. גם את קבצי ה-Excel ניצור באמצעות האתר [Mockaroo](#).

## אבלוס הטבלה Guest

באטר [Mockaroo](#) ניצור סכמה חדשה בשם Guest ונגידר עבורה שדות מתאימים.



בתחתית המסר נבקש שהנתונים יוצרו בפורמט SQL, כלומר קובץ SQL עם המונ פקודות `INSERT INTO`. את הקובץ נוריד ונשמר בשם `Guest.sql`.

## אבלוס הטבלה Room

באטר [Mockaroo](#) ניצור סכמה חדשה בשם Room ונגידר עבורה שדות מתאימים.

בתחתית המסך נבקש שהנתונים יוצאו בפורמט SQL, כלומר קובץ SQL עם המונח פקודות **INSERT INTO**. את הקובץ נוריד ונשמר בשם **Room.sql**.

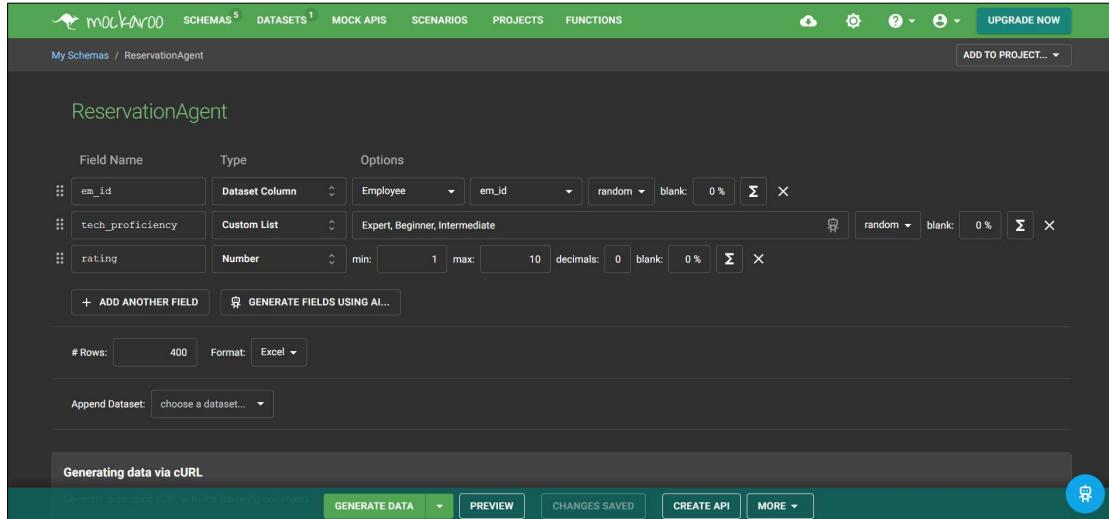
### אבלוס הטבלה Employee:

באטר **Mockaroo** נוצר סכמה חדשה בשם **Employee** ונגדיר עבורה שדות מתאימים.

את הפורט נבחר להיות CSV כיון שרק במקרה זה ניתן לשמור את הטבלה אצלו בתור **dataset**.  
בעת נוכל להיבנס לשוניית **DATASETS** ולהורית אותו שם בקובץ **.csv**. כיון ש-**SQL/PL** מאפשר לנו  
לייבא רק קבצי אקסל – נמיר את קובץ ה-**.csv**. לקובץ אסא. ונקרא לו **EmployeeTable.xlsx**.

## אבלוס הטבלה ReservationAgent

באטר *Mockaroo* ניצור סכמה חדשה בשם Employee ונגדיר עבורה שדות מתאימים. נשים לב שתת שדה em\_id נדרש ליחס ל-*id* של הסכמה Employee, וזאת הסיבה ששמרנו את הטבלה Employee שנוצרה, כ-set dataset באתה.

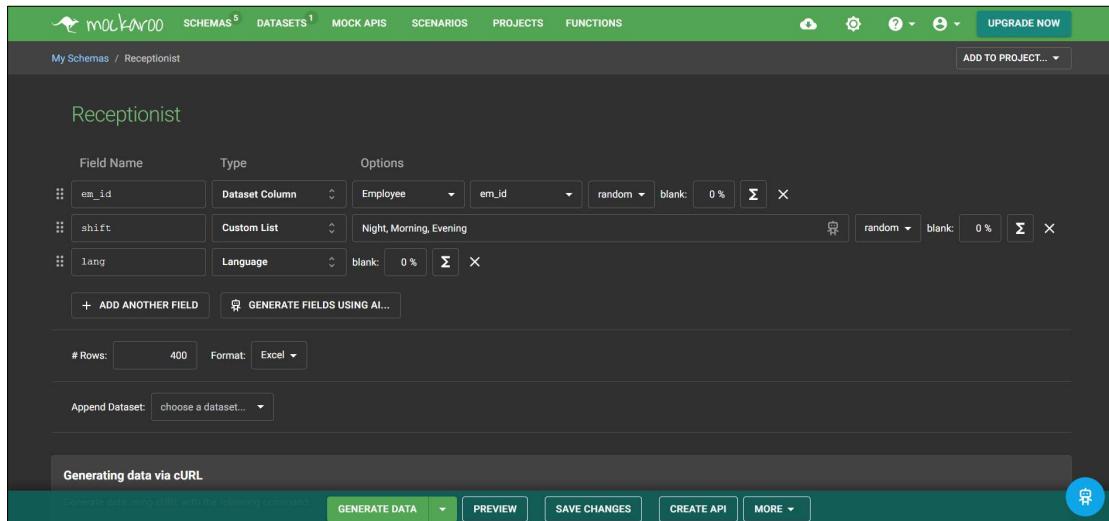


The screenshot shows the Mockaroo web application interface. At the top, there are navigation tabs: SCHEMAS (5), DATASETS (1), MOCK APIs, SCENARIOS, PROJECTS, and FUNCTIONS. On the right side, there are buttons for UPGRADE NOW, ADD TO PROJECT..., and other settings. The main area is titled "ReservationAgent". It displays three fields: "em\_id" (Dataset Column, Employee type, random, blank 0%), "tech\_proficiency" (Custom List, Expert, Beginner, Intermediate), and "rating" (Number, min 1, max 10, decimals 0, blank 0%). Below these are buttons for "+ ADD ANOTHER FIELD" and "GENERATE FIELDS USING AI...". There are also fields for "# Rows" (set to 400) and "Format" (set to Excel). A "Append Dataset" dropdown is present. At the bottom, there's a progress bar labeled "Generating data via cURL" and several buttons: GENERATE DATA, PREVIEW, CHANGES SAVED, CREATE API, and MORE.

בתחתית המסך נבקש שהנתונים יויצרו בפורמט קובץ Excel. את קובץ האקסל שנוצר נוריד ונשמור בשם `ReservationTable.xlsx`.

## אבלוס הטבלה Receptionist

באטר *Mockaroo* ניצור סכמה חדשה בשם Employee ו נגדיר עבורה שדות מתאימים. נשים לב שתת שדה em\_id נדרש ליחס ל-*id* של הסכמה Employee, וזאת הסיבה ששמרנו את הטבלה Employee שנוצרה, כ-set dataset באתה.

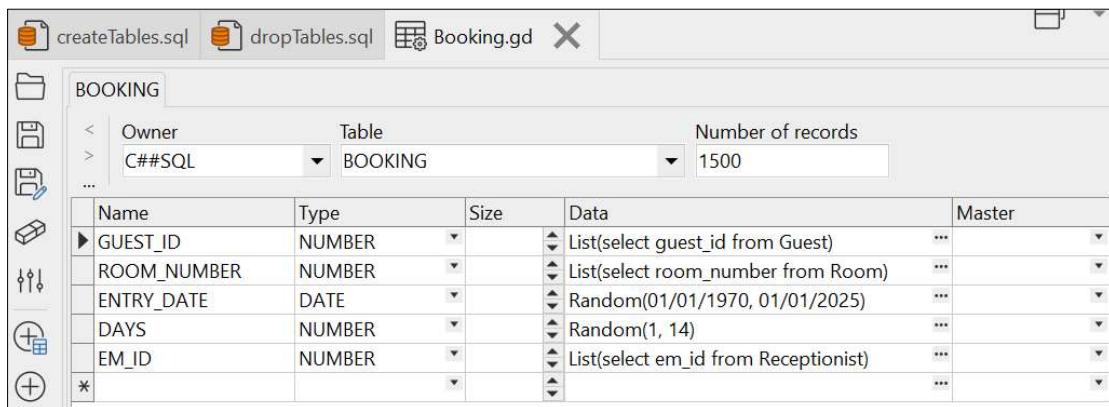


The screenshot shows the Mockaroo web application interface. At the top, there are navigation tabs: SCHEMAS (5), DATASETS (1), MOCK APIs, SCENARIOS, PROJECTS, and FUNCTIONS. On the right side, there are buttons for UPGRADE NOW, ADD TO PROJECT..., and other settings. The main area is titled "Receptionist". It displays three fields: "em\_id" (Dataset Column, Employee type, random, blank 0%), "shift" (Custom List, Night, Morning, Evening), and "lang" (Language, blank 0%). Below these are buttons for "+ ADD ANOTHER FIELD" and "GENERATE FIELDS USING AI...". There are also fields for "# Rows" (set to 400) and "Format" (set to Excel). A "Append Dataset" dropdown is present. At the bottom, there's a progress bar labeled "Generating data via cURL" and several buttons: GENERATE DATA, PREVIEW, SAVE CHANGES, CREATE API, and MORE.

בתחתיו המסר נבקש שהנתונים ייווצרו בפורמט קובץ Excel. את הקובץ האקסל שנוצר נוריד ונשמר בשם `ReceptionistTable.xlsx`.

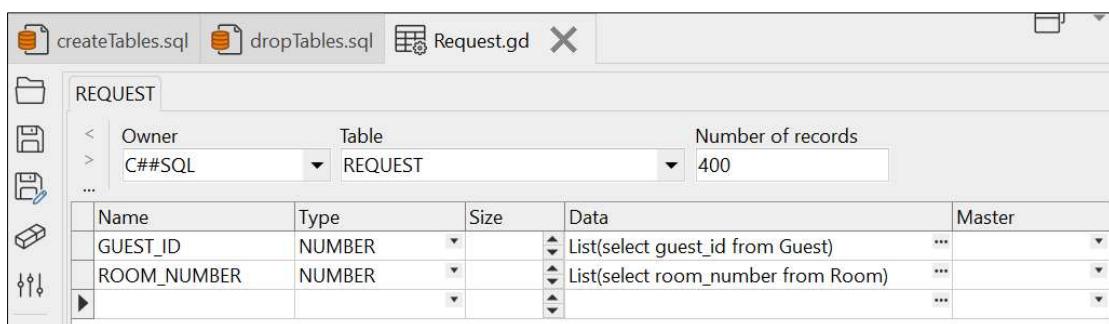
### אבלוס הטבלה :Booking

נשתמש בכלי Data Generator המובנה בתוך SQL/PL. את הקובץ ייצור נשמר בשם `Booking.gd`. כדי שנוכל להשתמש בו פעם נוספת אם נדרש.



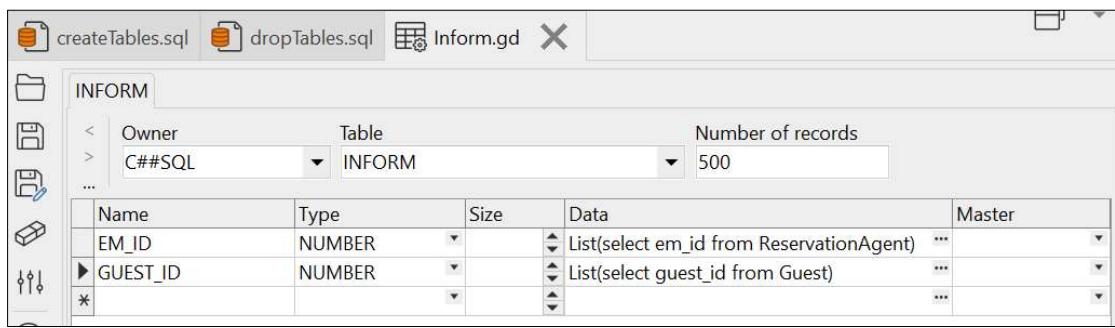
### אבלוס הטבלה :Request

נשתמש בכלי Data Generator המובנה בתוך SQL/PL. את הקובץ ייצור נשמר בשם `Request.gd`. כדי שנוכל להשתמש בו פעם נוספת אם נדרש.



### אבלוס הטבלה :Inform

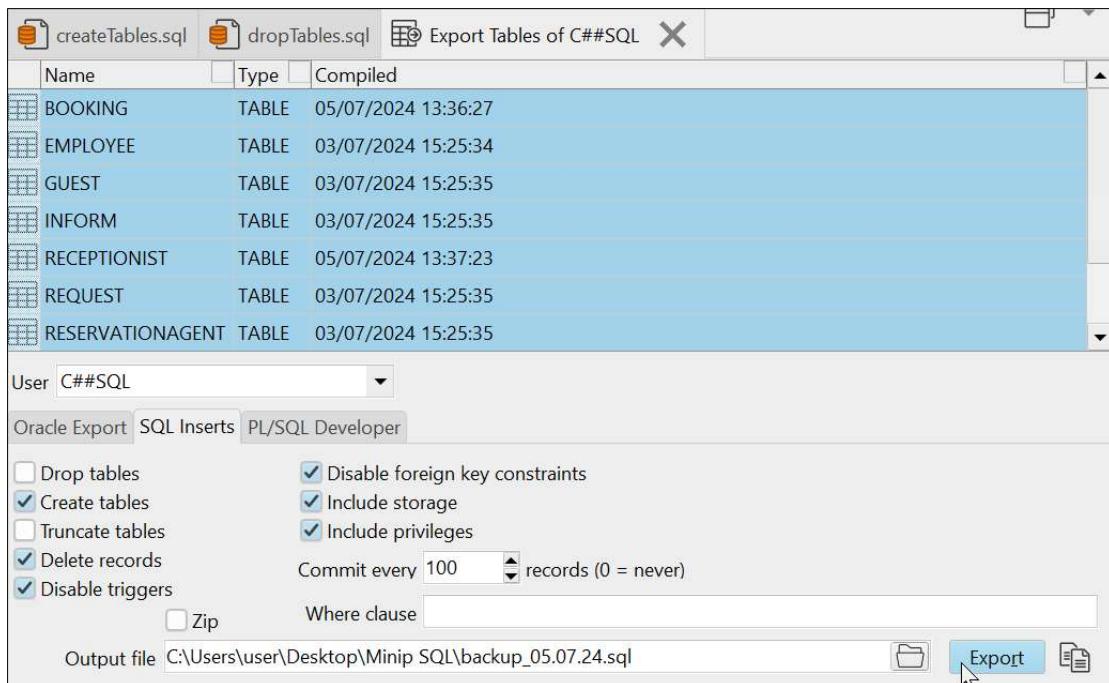
נשתמש בכלי Data Generator המובנה בתוך SQL/PL. את הקובץ ייצור נשמר בשם `Inform.gd`. כדי שנוכל להשתמש בו פעם נוספת אם נדרש.



## גיבוי ו恢復 בסיס הנתונים

### גיבוי בסיס הנתונים:

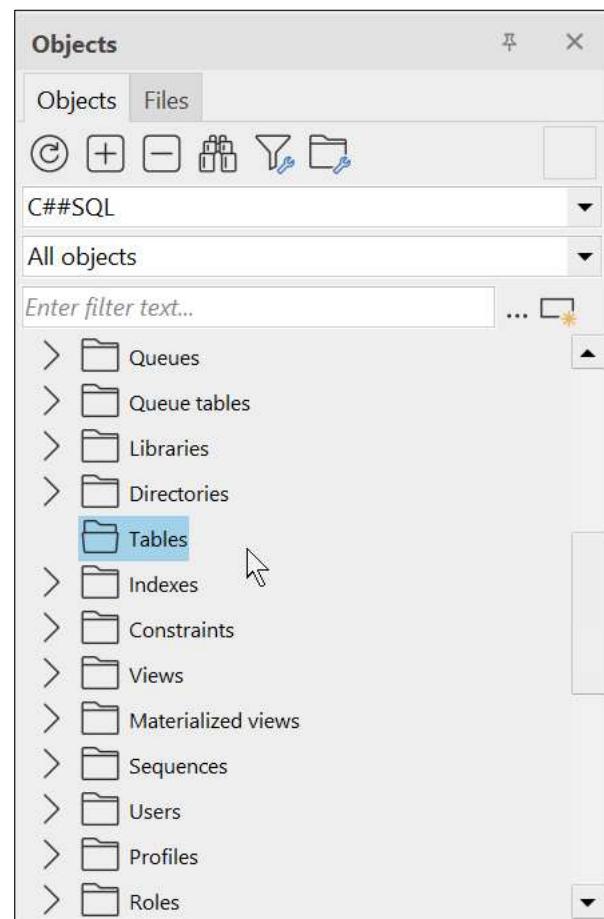
נגבא את כל נתונים ייצור ו記錄 הטבלאות שלנו אל תוך קובץ SQL ייעודי על מנת שנוכל לשחזר את הטבלאות או ליצור אותן על בסיס נתונים אחר מתי שנרצה.



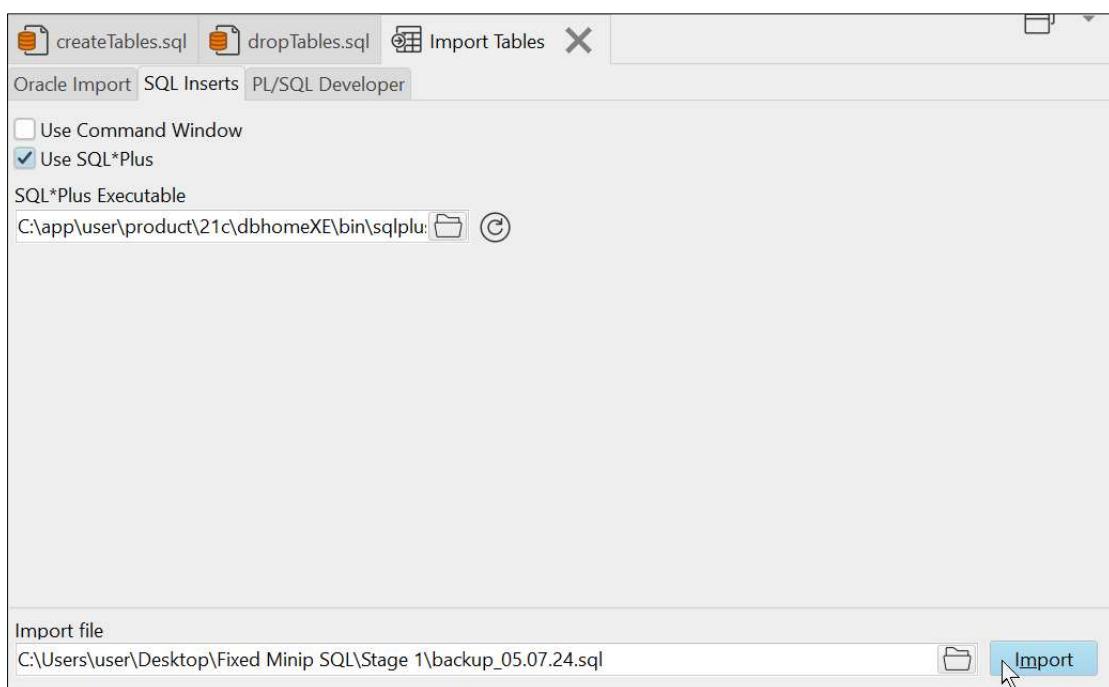
כל הקוד ליצירת בסיס הנתונים שלנו נמצא בקובץ backup\_05.07.24.sql

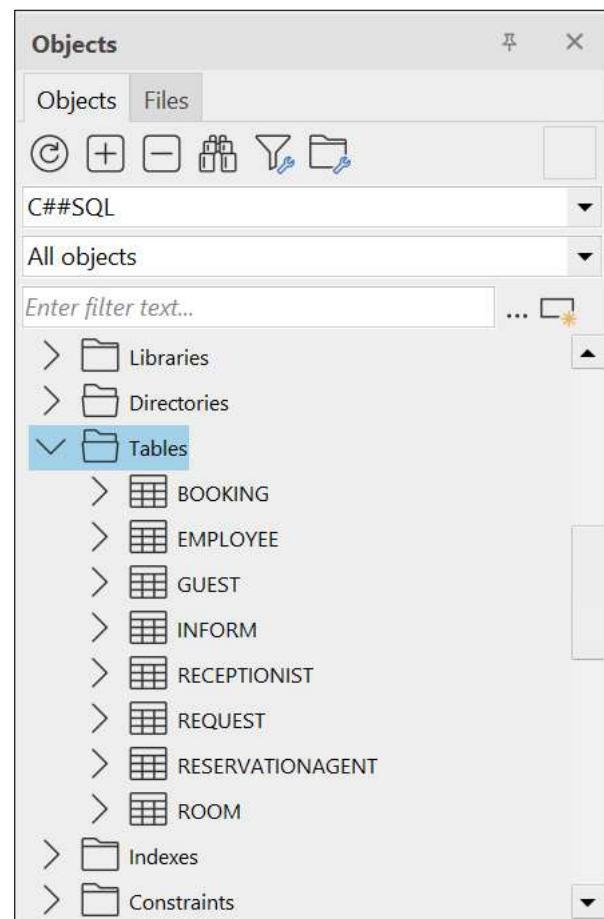
### ייבוא בסיס נתונים:

כאמור, נרצה ליבא את בסיס הנתונים שלנו או כי מחקנו אותו ונחנכו רצים לשחזר אותו, או אחרי שהחליט לטען אותו למערכת חדשה.



הרכינו את הקובץ qbq.s Import Tables- dropTables.sql וכל הטבלאות שלנו באמת נמחקו. נכנס ל- Import Tables וKİיבא את הקובץ הנ"ל.





נוכל לראות שאכן כל הטבלאות נוצרו ואוכלסו בתנאים הקודמים.

# שאילות על בסיס הנתונים

שאילות select:

שאילה 1:

הנהלת בית המלון קיבלה אישור ממשלה לבנות קומה נוספת לבית המלון. על מנת להחליט בצורה נכונה וכוננה ודויקת כמה מיטות עליהם להציב בכל חדר, הוחלט ללמידה משנים עברו אילו חדרים היו הפופולריים.

לצורך כך נכתב שאלתה שתמצאו לנו כמה אחוז מהמזינים הזמינים סוג חדר מסוים עבור כל אחד מסוגי החדרים השונים.

```
select
    beds,
    room_count / (
        select count(*)
        from Booking
        ) as frequency
from (
    select beds, count(*) as room_count
    from Room natural join Booking
    group by beds
    )
order by beds desc;
```

הרצת השאלתה:

The screenshot shows the MySQL Workbench interface with the following details:

- SQL Editor:** Contains the SQL query provided above.
- Output Tab:** Displays the results of the query in a grid format.
- Results Grid:** Shows the following data:

	BEDS	FREQUENCY
1	5	0.3313333333333333
2	4	0.344
3	2	0.3246666666666667

## שאילהה 2:

לאחר ביקורת של משרד התברואה בבית המלון הtagלו במספר חדרים במלון חורים זעירים בקיר שבין החדר למרפסת החדר. החורים גרמו לעכברים שהגיבו מבחן לאחיכנס אל תוך החדר, ולהפריע את מנוחתם של האורחים.

הנהלת בית המלון החליטה לפצות את כל האורחים שהיו בחדרים 304, 112, 466, בשובר לסייע פיצה בנו פלאנט שנסגרה לפני שנתיים. לצורך כך נכתב שאלתה שתאתר לנו את כל אוטם האורחים.

```
select
    first_name,
    last_name,
    phone
from
    Booking natural join Guest
where
    entry_date < date '2007-06-01'
    and room_number in (112, 304, 466);
```

הרצת השאלתה:

The screenshot shows the MySQL Workbench interface. In the top tab bar, there are three tabs: 'createTables.sql', 'dropTables.sql', and 'select first\_name, last\_na ...'. The third tab is active. Below the tabs, there are three buttons: 'SQL', 'Output', and 'Statistics', with 'SQL' being the active one. The main area contains the SQL query:

```
select
    first_name,
    last_name,
    phone
from
    Booking natural join Guest
where
    entry_date < date '2007-06-01'
    and room_number in (112, 304, 466);
```

Below the query, there is a toolbar with various icons for database management. The bottom half of the window displays the results of the query in a grid format:

	FIRST_NAME	LAST_NAME	PHONE
1	Teador	Gravet	056-4714271
2	Beverley	Howey	051-2526042
3	Marven	Reddish	057-9705310
4	Dud	Trass	056-6348136
5	Mady	Westmerland	052-8001091

## שאילהה 3:

בעקבות התכנון לבניית קומה נוספת לבנייה נספה לבניית המון נדרשת המון עזרה במלצות השונות הרכומות במשימה. בית המלון הגיע למסקנה שהעובדים הבכירים יוכננים בתפקוד התקין של בית המלון הם

הסוכנים, הדלקאים וצוות הנהלה העליון של המלון, וכן הם יתמידו בתפקידם. כל מי שאינו מהמנויים לעיל נקרא לדגל ויסיע בכל מה שצריך.

נכתב שאלתה שתמצא לנו את כל העובדים שנדרשים להירთ למשימה, בהנחה שהעובדים בצוות הנהלה העליון מקבלים משכורת חדש של מעל ₪14,000.

```
select *
from Employee
where em_id not in (
    select em_id
    from ReservationAgent
    union
    select em_id
    from Receptionist
)
and salary <= 14000;
```

הרצת השאלתה:

The screenshot shows a SQL editor window with three tabs at the top: 'createTables.sql', 'dropTables.sql', and 'select \* from Employee whe ...'. The active tab contains the following SQL code:

```
select *
from Employee
where em_id not in (
    select em_id
    from ReservationAgent
    union
    select em_id
    from Receptionist
)
and salary <= 14000;
```

Below the code is a results grid displaying the following data:

	EM_ID	FIRST_NAME	LAST_NAME	SALARY	
▶	1	867142561	Lilian	Beardsdale	13070
	2	985038742	Alyssa	McCuish	7502
	3	601965364	Stephen	Volkes	10610
	4	889749658	Gleda	Banisch	13372
	5	935283934	Ronna	Gostall	13132
	6	504063163	Barnabe	Wilmore	11827

#### שאילה 4:

בבית המלון החליטו להעניק מתנה לאורחים שבאים לפוש בחגים (למנין ל"ע). המתנה תהיה בקבוק יין גסך עטוף בצלפון על מצע עור חזיר.

נכתב שאלתה שתמצא לנו את כל ההזמנות לחג המולד, ראש השנה האזרחי, יום האהבה וחג הפסחא, ותמיין אותם על פי סדר התאריכים.

```
select *
from Booking
```

```

where (extract(day from entry_date) = 1 and extract(month from
entry_date) = 1)
    or (extract(day from entry_date) = 25 and extract(month from
entry_date) = 12)
    or (extract(day from entry_date) = 14 and extract(month from
entry_date) = 2)
order by entry_date;

```

הרצת השאלתה:

The screenshot shows a MySQL Workbench window with three tabs at the top: 'createTables.sql', 'dropTables.sql', and 'select \* from Booking wher ...'. The 'SQL' tab is selected, displaying the query:

```

select *
from Booking
where (extract(day from entry_date) = 1 and extract(month from entry_date) = 1)
    or (extract(day from entry_date) = 25 and extract(month from entry_date) = 12)
    or (extract(day from entry_date) = 14 and extract(month from entry_date) = 2)
order by entry_date;

```

Below the SQL tab is a toolbar with various icons for database management. The main area contains a results grid with the following data:

	GUEST_ID	ROOM_NUMBER	ENTRY_DATE	END_DATE	DAYS	EM_ID
▶	1	814830056	684 25/12/1970	...	6	324539265
	2	412041638	243 25/12/1971	...	13	953949724
	3	235322446	430 25/12/1975	...	6	432069073
	4	155712522	113 14/02/1976	...	6	701956875
	5	513883630	576 25/12/1982	...	10	796939742
	6	43674898	632 25/12/1991	...	6	976997166
	7	306494187	183 25/12/2006	...	7	948482928

## שאילות delete:

### שאילתת 1:

איציק עובד האחזקה טיפל היום בכל הבקשות שהתקבלו מחדר 205. הוא מעוניין להסיר את הבקשות מהמערכת על מנת שלא יטעו לחשב שהן עוד קיימות.

כתבו שאלתה שתסיר מטבלת הדיווחים את הרשומות שמייחסות לחדר 205.

```

delete
from Request
where room_number = 205;

```

לפני הרצת השאלתה:

```

SQL Output Statistics
select *
from Request
where room_number = 205;

```

	GUEST_ID	ROOM_NUMBER
1	595423656	205
2	885530745	205

אחרי הרצת השאלתה:

```

SQL Output Statistics
select *
from Request
where room_number = 205;

```

	GUEST_ID	ROOM_NUMBER

## שאילתה 2:

לאחרונה הגיעו מספר עובדים שטענו שהם לא מספקים להיות עם המשפחה שלהם בעבריהם והידלה הקטנה בוכה בלילה והכל מופל על האישה. לאחד בירור עמוק הנהלת בית המלון גילתה שאוטם העובדים הם גם סוכנים וגם דלפקאים.

כיוון שהנהלת בית המלון מעודדת שוויון בנטול וגם הבעלים צריכים לקיים באמצעות הלילה לידה, הוחלט פה אחד שככל עובד שהוא גם סוכן וגם דלקאי מעתה לא יהיה סוכן כי יש יותר מדי כאלה.

נכתב שאיילתה שתמוך מבסיס הנתונים את הכפיפות האלה ותעדכן את הטבלה של הסוכנים.

```
delete
from ReservationAgent
where em_id in (
    select em_id
    from ReservationAgent
    intersect
    select em_id
    from Receptionist
);

```

לפני הרצת השאלה:

The screenshot shows the MySQL Workbench environment. In the top tab bar, there are three tabs: 'createTables.sql', 'dropTables.sql', and 'select \* from ReservationA ...'. The 'select \* from ReservationA ...' tab is active, showing the SQL query provided above. Below the tabs is a toolbar with various icons for database management. The main area contains a results grid with the following data:

	EM_ID	TECH_PROFICIENCY	RATING
▶	1	838539231 Beginner	1
2	422957628 Expert	10	
3	885641852 Beginner	9	
4	894962236 Beginner	10	
5	501487636 Expert	3	
6	794922762 Intermediate	9	

אחרי הרצת השאלה:

```

SQL Output Statistics
select *
from ReservationAgent
where em_id in (
    select em_id
    from ReservationAgent
    intersect
    select em_id
    from Receptionist
);

```

EM_ID	TECH_PROFICIENCY	RATING

## שאילות update:

### שאילה 1:

גברת קרש ומר מערוך הזמין חדר בבית המלון לתאריך 01/09/2024. לאחר ביקור אצל רופא המשפחה נודע למර מערוך שהוא סובל מבקעת חמורה בכל חלקו מערכו, אך שיצטרך לבצע ניתוח להסרת בזק בלילה. לחו מזמן החלטמה אחורי נופלים על זמן החופשה שקבעו.

גברת קרש פנתה לヨיסי הדלקאי שמתפל בהם בבקשת לדוחות את מועד ההזמנה בשבועיים. שם הבמה של גברת קרש הוא Trixie Punter ועליה רשמה את הזמנתה.

כתבו שאלה שתעדכן את התאריך החדש שיתאים לאב המשפחה המחלים.

```

update Booking
set entry_date = date '2024-09-30'
where guest_id =
    (select guest_id
     from Guest
     where first_name = 'Trixie' and last_name = 'Punter')
and entry_date > date '2024-07-07';

```

לפני הרצת השאלה:

The screenshot shows a SQL development environment with three tabs at the top: "createTables.sql", "dropTables.sql", and "select \* from Booking wher...". The active tab is "SQL". The query in the SQL pane is:

```
select *
from Booking
where guest_id = (
    select guest_id
    from Guest
    where first_name = 'Trixie' and last_name = 'Punter'
)
and entry_date > date '2024-07-07';
```

The results pane below displays a single row of data:

	GUEST_ID	ROOM_NUMBER	ENTRY_DATE	DAY	EM_ID
▶ 1	315806874	223	01/09/2024	...	13 440327900

אחרי הרצת השאלתה:

The screenshot shows the same SQL development environment. The query in the SQL pane is identical to the one above:

```
select *
from Booking
where guest_id = (
    select guest_id
    from Guest
    where first_name = 'Trixie' and last_name = 'Punter'
)
and entry_date > date '2024-07-07';
```

The results pane displays a single row of data:

	GUEST_ID	ROOM_NUMBER	ENTRY_DATE	DAY	EM_ID
▶ 1	315806874	223	30/09/2024	...	13 440327900

## שאילתה 2:

לבבוד יומם העובד החירוז החלטה הנהלת בית המלון להעלות את שכרם החודשי של סוכנים אשר מספר האורחים שיצרו איתם קשר לבירור עולה על שלושה. הם יקבלו תוספת קבועה של ₪1000 במשכורותם החודשיות.

כתבו שאלה שתמצא לנו את אותם סוכנים ותעדכן את משכורתם בהתאם.

```

update Employee
set salary = salary + 1000
where em_id in (
    select em_id
    from (
        select em_id, count(*) as guests
        from inform
        group by em_id
    )
    where guests > 3
);

```

לפני הרצת השאלה:

The screenshot shows a SQL interface with three tabs at the top: 'createTables.sql', 'dropTables.sql', and 'select \* from Employee nat ...'. The active tab contains the following SQL code:

```

select *
from Employee
natural join (
    select em_id, count(*) as guests
    from inform
    group by em_id
)
where guests > 3;

```

Below the code is a toolbar with various icons. The main area displays a table with the following data:

	EM_ID	FIRST_NAME	LAST_NAME	SALARY	GUESTS	
▶	1	285417351	Mina	Glenny	8395	5
	2	73319693	Roth	Cowden	9170	4
	3	770042193	Pearce	Ianinotti	8654	4
	4	942345154	Josepha	Lunt	9529	5
	5	350875712	Ruperta	Balm	15867	4
	6	646600561	Dick	McIlhagga	19761	4
	7	55152143	Demetria	Seldon	14592	4
	8	852821102	Katie	Padula	14655	4

אחרי הרצת השאלה:

The screenshot shows a SQL development environment with the following details:

- Top Bar:** Three tabs are visible: "createTables.sql", "dropTables.sql", and "select \* from Employee nat ...".
- Toolbar:** Standard SQL editor toolbar with icons for new file, save, cut, copy, paste, etc.
- Code Area:** A code editor containing the following SQL query:
 

```
select *
from Employee
natural join (
    select em_id, count(*) as guests
    from inform
    group by em_id
)
where guests > 3;
```
- Results Grid:** A table displaying the results of the query. The columns are labeled: EM\_ID, FIRST\_NAME, LAST\_NAME, SALARY, and GUESTS. The data consists of 8 rows:

	EM_ID	FIRST_NAME	LAST_NAME	SALARY	GUESTS	
▶	1	285417351	Mina	Glenny	9395	5
	2	73319693	Roth	Cowden	10170	4
	3	770042193	Pearce	Ianinotti	9654	4
	4	942345154	Josepha	Lunt	10529	5
	5	350875712	Ruperta	Balm	16867	4
	6	646600561	Dick	McIlhagga	20761	4
	7	55152143	Demetria	Seldon	15592	4
	8	852821102	Katie	Padula	15655	4

## שאילות עם פרטיטורים:

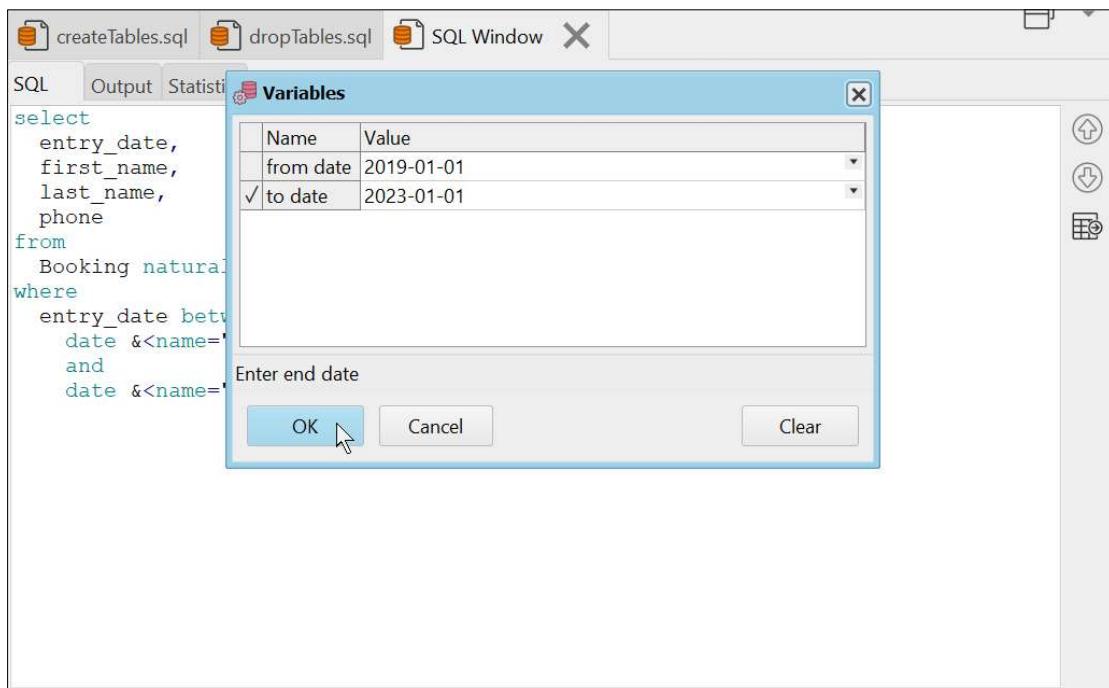
### שאילה 1:

אדון זהבי ואשתו לקחוות קבועים בבית המלון. כשהזרו הביתה אחרי החופשה האחורה שלהם הייתה בהחינה במלון, שמו לב לפטע שהסוכירה על מקל שהזמין ל��וד הרាជון מאמזון אבדה ככל הנראה בזמן שהותם בבית המלון. גברת זהבי חושדת שהסוכירה נפלה לה בזמן שהלבה לאוטו עם ערמה של תיקים בידיהם, ביום שבו עזבו את בית המלון, לפני עשרה ימים.

היא פנתה לבית המלון בבקשת שיבדקו במצולמות האבטחה, אך לרווח מזלה מצולמות האבטחה בקומת זו לא היו תקינות. גברת זהבי החליטה ליצור קשר עם כל אחד מהאורחים בקומת 4, שם ככל הנראה נראתה הסוכירה לאחרונה. לשם כך היא ביקשה מהמלון את פרטיים של כל האורחים שנכחו בקומת באותם עשרה ימים.

```
select
    entry_date,
    first_name,
    last_name,
    phone
from
    Booking natural join Guest
where
    entry_date between
        date &<name="from date" hint="Enter start date" type=string>
        and
```

```
date &<name="to date" hint="Enter end date" type=string>
```



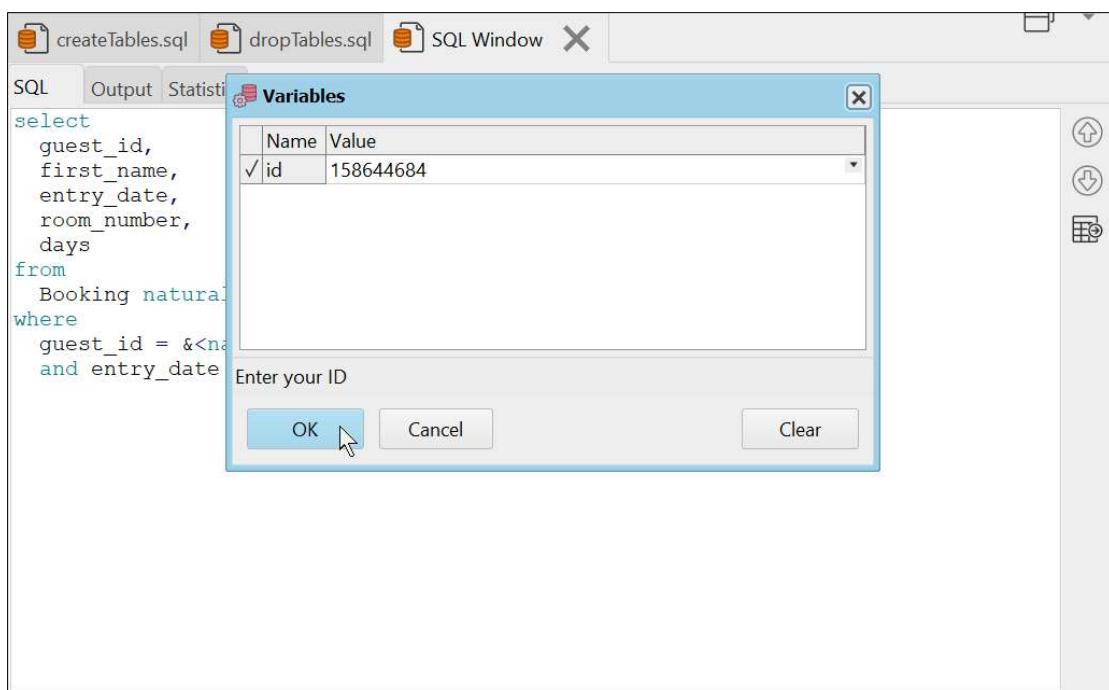
	ENTRY_DATE	FIRST_NAME	LAST_NAME	PHONE
1	09/02/2020	Alonso	McMenamy	052-4280703
2	24/03/2019	Fremont	Clawley	050-3282462
3	14/01/2019	Tamar	Tunsley	053-4434401
4	30/10/2022	Petey	Magrannell	051-8643850
5	02/11/2019	Chalev	Dallocchi	057-52815690

## שאילה 2:

mbin האורחים של בית המלון ישנים כמעט מאד מפוזרים, ברמה זאת שם עלולים לשכוח למתי הזמין חדר בבית המלון, אם בכלל. כדי למנוע עוגמת נפש, בית המלון החליט לפתח ללקחות שלן את האפשרות להתעדכן שוב בהזמנות העתידיות שלהם לבית המלון.

נכטוּ שאיילתָה שתקבל מהמשתמש את תעודת הזיהות שלו (המזהיּוֹן) ותחזיר לו את פירוט ההזמנות העתידיות שלו.

```
select
    guest_id,
    first_name,
    entry_date,
    room_number,
    days
from
    Booking natural join Guest
where
    guest_id = &<name=id hint="Enter your ID">
    and entry_date > date '2024-01-01';
```



The screenshot shows a SQL development environment. In the top tab bar, there are three tabs: 'createTables.sql', 'dropTables.sql', and 'select guest\_id, first\_name ...'. The active tab is 'SQL'. Below the tabs, there is a toolbar with various icons for database operations like insert, update, delete, and search. The main area contains a SQL query:

```

select
    guest_id,
    first_name,
    entry_date,
    room_number,
    days
from
    Booking natural join Guest
where
    guest_id = &<name=id hint="Enter your ID">
    and entry_date > date '2024-01-01';

```

Below the query, the results are displayed in a grid:

	GUEST_ID	FIRST_NAME	ENTRY_DATE	ROOM_NUMBER	DAYS	
▶	1	158644684	Marven	16/06/2024	592	13

### שאילתת 3:

עובד התחזקה של בית המלון עבדים קשה משבב לשעון כדי לטפל בכל תקליה בהקדם האפשרי. לאורחיו המלון יש את האפשרות לדוח על תקלות בחדרי המלון, כדי שצוות התחזקה ידע על קיומן.

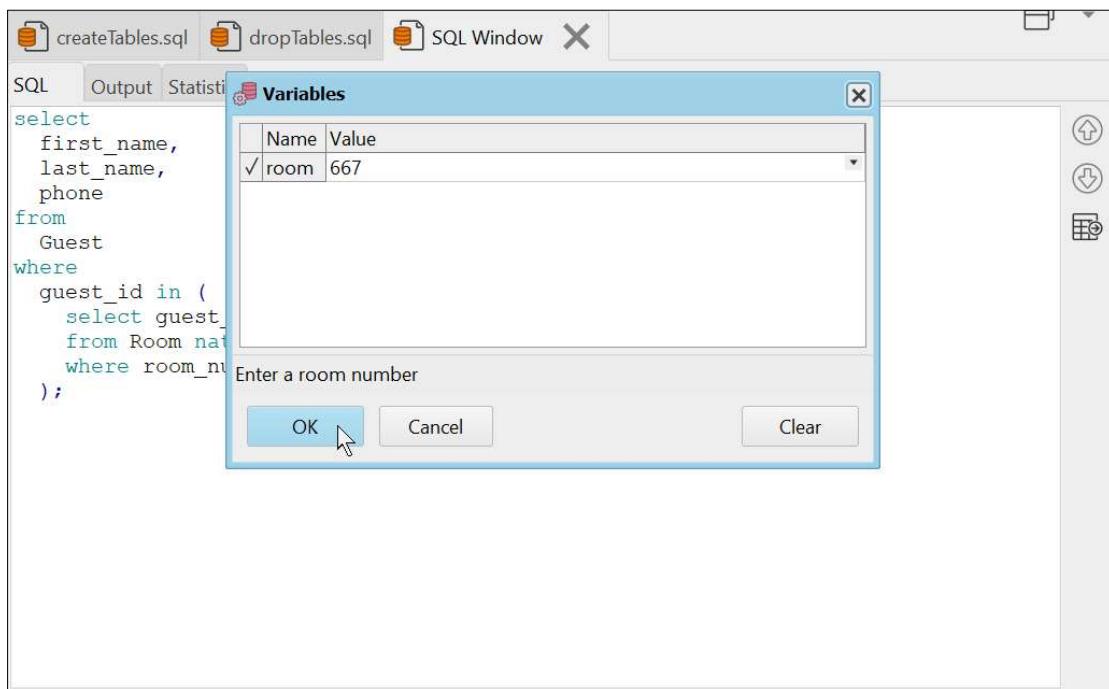
עובד ניוקין יוכל אם ירצה לשאול עבור חדר מסוים מה התקנות שדוחו לגביו. כדי לקבל תשובה על שאלה זו יידרש העובד לפרט הקשר של המדווחים כדי לשמוע על התקלה המקורי ראשון.

כתבו שאלתה עבור עובד בית מלון שיספק שם של חדר ויקבל את הפרטים של האנשים שדווחו עליו אותו חדר.

```

select
    first_name,
    last_name,
    phone
from
    Guest
where
    guest_id in (
        select guest_id
        from Room natural join Request
        where room_number = &<name=room hint="Enter a room number">
    );

```



	FIRST_NAME	LAST_NAME	PHONE
1	Riccardo	Minghi	051-5944580
2	Mariann	Metzing	056-9103918
3	Lloyd	Asple	053-2635718

#### שאילתה 4:

כאשר לקוח נכנס לאתר ההזמנות של בית המלון, עליו לבחור חדר. הלקוח יודע כמה מיטות הוא מעוניין שייהי בחדר, אך הוא לא יודע אילו חדרים עונים על הדרישת שלו.

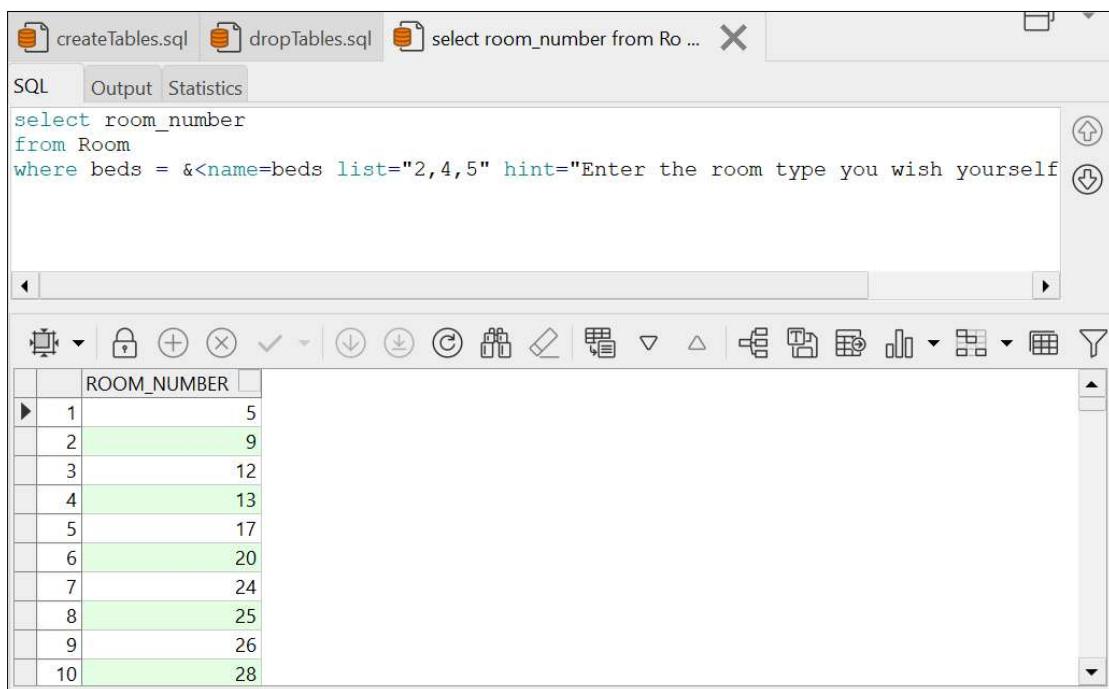
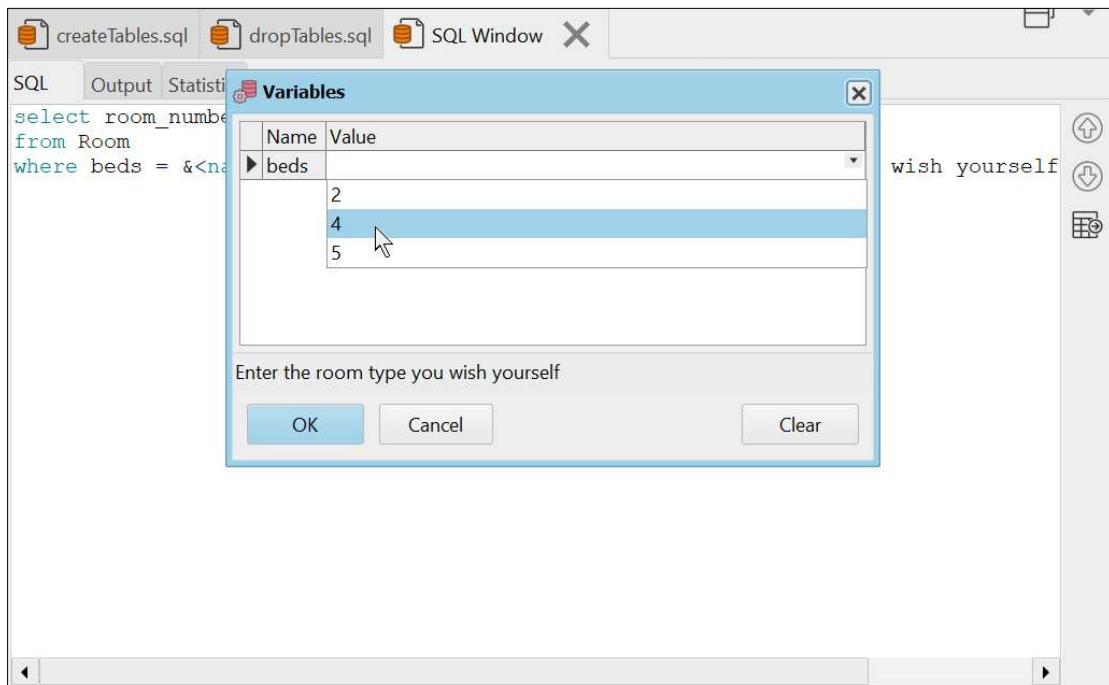
נכתב שאליה שבה המזמין יבחר את סוג החדר שירצה (לפי מספר המיטות) מתוך רשימה, ויקבל בחזרה את רשימת כל החדרים במלון מסווג זה.

```
select room_number
```

```

from Room
where beds = &<name=beds list="2,4,5" hint="Enter the room type you
wish yourself" type=integer>;

```



## הוספת אילוצים לטיבלאות:

### אילוץ ראשון:

בשאלתת המחיקה השנייה ניסינו למחוק מספר סוכנים וזה לא התאפשר כיון שהם היו משוייכים לרשומות Inform מסוימות. נסיף בטבלה Inform אילוץ on delete cascade לשדה em\_id, ואז כשןמחק סוכן מסוים – כל הרשומות המשויכות אליו תמחקנה גם.

```
alter table Inform
drop constraint fk_em_id_inform;

alter table Inform
add constraint fk_em_id_inform
foreign key (em_id) references ReservationAgent (em_id) on delete
cascade;
```

נראה בנתוני הטבלה Inform שבאמת נוסף האילוץ שרצינו:

Name	Type	Columns	Enabled	Referencing table	Referencing columns	On Delete	Deferrable
SYS_C009128	Primary	EM_ID, GUEST_ID	<input checked="" type="checkbox"/>				<input type="checkbox"/>
► FK_EM_ID_INFORM	Foreign	EM_ID	<input checked="" type="checkbox"/>	RESERVATIONAGENT	EM_ID	Cascade	<input type="checkbox"/>
SYS_C009130	Foreign	GUEST_ID	<input checked="" type="checkbox"/>	GUEST	GUEST_ID	No action	<input type="checkbox"/>

### אילוץ שני:

נסיף אילוץ על הערך של salary בטבלה Employee כדי לוודא שהמשכורת שהוזנה נמצאת בתחום התקין.

```
alter table Employee
add constraint salary_check check (salary between 5000 and 30000);
```

כעת נראה שבאמת לא ניתן להוסיף רשותה שחורגת מההגבלות:

The screenshot shows an Oracle SQL Developer interface. In the SQL tab, there is a script window containing the following SQL code:

```

alter table Employee
add constraint salary_checking check (salary between 5000 and 30000);

insert into Employee (em_id, first_name, last_name, salary)
values (123, 'ami', 'breni', 50000);

```

An error dialog box is displayed, showing the message: "ORA-02290: check constraint (C##SQL.SALARY\_CHECKING) violated". The dialog has three buttons: OK, Cancel, and Help.

Below the SQL tab, the Output tab shows the message "(no result set)".

### אילוץ שלישי:

נוסף אילוץ לשדה lang שבטבלה Receptionist על מנת שערך ברירת המחדל של השפה בה יתקשר המארח תהיה אנגלית.

```
alter table Receptionist
modify lang default 'English';
```

נראה שההשאולתה הצלחה:

The screenshot shows an Oracle SQL Developer interface. In the SQL tab, there is a script window containing the following SQL code:

```

alter table Receptionist
modify lang default 'English';

insert into Receptionist (em_id, shift)
values (867142561, 'Night');
commit;

select lang
from Receptionist
where em_id = 867142561;

```

Below the SQL tab, the Output tab shows the results of the select query:

LANG
1 English

# תכנות ב- PL/SQL

## פונקציות:

### פונקציה ראשונה:

המנהל החדש רון רצה לוודא שהמחירים שהוא עברו החדרים תואמים את הסטנדרטים הגבוהים של המלון. כדי לעשות זאת, הוא מזקק לדרך קלה ומהירה לחשב את המחיר הממוצע למיטה בכל חדרי המלון.

רון פנה לדן, ובקש ממנו למצאו פתרון. דן כתוב פונקציה שנקראת `bed_average_price`, שמטරתה לחשב את המחיר הממוצע של מיטה בכל חדרי המלון.

### קוד לייצרת הפונקציה:

```
create or replace function bed_average_price return number
as
    sum_of_beds number := 0;
    total_prices number := 0;
    average_of_price_for_bed number;
begin
    -- Sum up total price and number of beds from Room table
    select sum(price), sum(beds)
    into total_prices, sum_of_beds
    from Room;

    -- Calculate and round average price per bed
    average_of_price_for_bed := round(total_prices/sum_of_beds, 2);

    return (average_of_price_for_bed);
end;
```

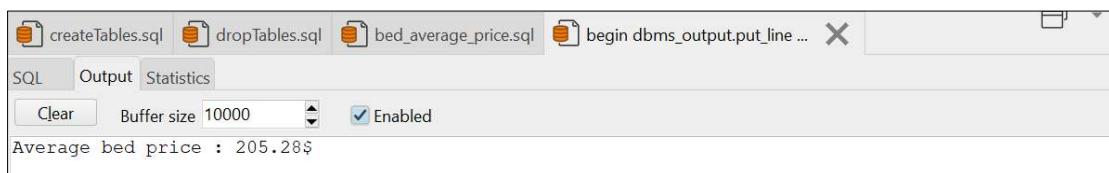
נ裏ץ את התוכנית הבאה לדוגמה:



The screenshot shows the Oracle SQL Developer interface. In the top tab bar, there are four tabs: 'createTables.sql', 'dropTables.sql', 'bed\_average\_price.sql', and 'begin dbms\_output.put\_line ...'. The current tab is 'bed\_average\_price.sql'. Below the tabs, there are three tabs: 'SQL' (selected), 'Output', and 'Statistics'. In the main SQL editor area, the following PL/SQL code is written:

```
begin
    dbms_output.put_line('Average bed price : ' || bed_average_price || '$');
end;
```

הפלט שמתתקבל הוא:



The screenshot shows the Oracle SQL Developer interface with the 'Output' tab selected. At the top, it shows the same four tabs as the previous screenshot. Below the tabs, there are three buttons: 'Clear', 'Buffer size 10000', and a checked checkbox labeled 'Enabled'. The main output window displays the following text:

```
Average bed price : 205.28$
```

## פונקציה שנייה:

המנהל החדש של בית המלון, רון, נכנס לתפקידו בשיא העונה ומיד החל בשינויים ושיפורים שראא לנגדון. הוא התמודד עם בעיה לא פשוטה, כמעט כל חדרי המלון תפוסים. רון רצה לוודא שתמיד יוכל לאורחים חדשים קומה עם חדרים פנויים ככל האפשר, במיוחד כאשר יש קבוצות גדולות המגיעות יחד.

רון פנה לתוכנת המלון, דן, וביקש ממנו פתרון. דן כתוב פונקציה שמצויה למצוא את הקומה שבה יש הכי הרבה חדרים פנויים, בהתבסס על תאריך הכניסה של הקבוצה ומספר הימים שהוא מתכנסת לשוהות במלון. הפונקציה שכותב נקראת `find_most_empty_floor`.

לצורך פישוט התהילה, ניצור פונקציית עזר בשם `find_floor` שתקבל מספר חדר ותחזיר את מספר הקומה בה הוא נמצא. הקוד לייצורה הוא:

```
create or replace function find_floor(room number) return number
as
begin
    return floor(room / 100);
end;
```

## הקוד לייצורת הפונקציה:

```
create or replace function find_most_empty_floor(p_group_entry_date
date, p_days number) return number
as
v_most_empty_floor number := null;
v_max_rooms number := 0;
v_current_floor number;
v_room_count number;
type t_room_list is table of number;
v_available_rooms t_room_list;

begin
    -- Try block
begin
    -- get available rooms
    select room_number
    bulk collect into v_available_rooms
    from Booking b1
    where b1.room_number not in (
        select b2.room_number
        from Booking b2
        where (b2.entry_date <= p_group_entry_date + p_days - 1
            and b2.entry_date + b2.days - 1 >= p_group_entry_date)
    );
    -- find floor with most available rooms
    for i in 1..v_available_rooms.count loop
        v_current_floor := find_floor(v_available_rooms(i));
        if v_current_floor > v_max_rooms then
            v_max_rooms := v_current_floor;
            v_most_empty_floor := i;
        end if;
    end loop;
end;
end;
```

```

-- count rooms on current floor
v_room_count := 0;
for j in 1..v_available_rooms.count loop
    if find_floor(v_available_rooms(j)) = v_current_floor then
        v_room_count := v_room_count + 1;
    end if;
end loop;

-- update most empty floor if necessary
if v_room_count > v_max_rooms then
    v_max_rooms := v_room_count;
    v_most_empty_floor := v_current_floor;
end if;
end loop;

-- Throw an exception if v_most_empty_floor is null
if v_most_empty_floor is null then
    raise_application_error(-20001, 'No empty floor found');
end if;

return v_most_empty_floor;

-- Catch block
exception
    when no_data_found then
        dbms_output.put_line('No data found');
        return null;
    when others then
        dbms_output.put_line('Error in find_most_empty_floor: ' ||
sqlerrm);
        return null;
    end;
end;

```

נ裏 את התוכנית הבאה לדוגמה:

```

begin
    dbms_output.put_line('Recommended floor: ' || find_most_empty_floor(date '1990-01-01', 12));
end;

```

הפלט שמתקיים הוא:

createTables.sql	dropTables.sql	find_most_empty_floor.sql	begin dbms_output.put_line ...	X
SQL	Output	Statistics		
<input type="button" value="Clear"/> Buffer size 10000 <input type="checkbox"/> Enabled Recommended floor: 2				

## פרוצדורות:

### פרוצדורה ראשונה:

לローン ולצאות הנהלה הישן, היה חשוב להבטיח שכל אורח יתקבל בצורה הטובה ביותר ביותר על ידי פקיד קבלה שדובר את שפתו של האורח ושהוא זמין בזמן הצ'ק-אין של האורח. רון פנה שוב לדן, המתכונת המוכשר של המלון, וביקש ממנו לעזור לו למצאו פתרון לבעה זו.

דן פיתח פרוצדורה בשם assign\_receptionist שהנועדה להקצת פקיד קבלה לכל אורח בהתבסס על שפת האורח ושעה הצ'ק-אין שלו.

### הקוד ליצירת הפרוצדורה:

```
create or replace procedure assign_receptionist(
    g_id number,
    check_in_time number,
    guest_lang varchar2
)
as
    check_shift varchar2(10);
    assigned_receptionist number;

begin
    -- determine shift based on check-in time
    if check_in_time >= 7 and check_in_time <= 15 then
        check_shift := 'Morning';
    elsif check_in_time >= 16 and check_in_time <= 23 then
        check_shift := 'Evening';
    else
        check_shift := 'Night';
    end if;

    begin
        -- try to find receptionist with matching language and shift
        select em_id
        into assigned_receptionist
        from Receptionist
        where lang = guest_lang
            and shift = check_shift
            and rownum = 1;
    exception
        when no_data_found then
            -- if not found, assign random receptionist with matching
language
            select em_id
            into assigned_receptionist
            from (
                select em_id
                from Receptionist
                where lang = guest_lang
                order by dbms_random.value
            )
            where rownum = 1;
    end;
end;
```

```

-- update assigned receptionist's shift
update Receptionist
set shift = check_shift
where em_id = assigned_receptionist;

dbms_output.put_line('receptionist ' || assigned_receptionist
|| ' moved to ' || check_shift || ' shift.');
end;

-- raise error if no receptionist found
if assigned_receptionist is null then
    raise_application_error(-20001, 'no receptionist available for
the specified language.');
end if;

-- assign receptionist to guest
update Booking
set em_id = assigned_receptionist
where guest_id = g_id;

dbms_output.put_line('guest ' || g_id || ' assigned to receptionist
' || assigned_receptionist || '.');

exception
when others then
    dbms_output.put_line('an error occurred: ' || sqlerrm);
end;

```

לפני הרצת הפרוצדורה:

The screenshot shows the Oracle SQL Developer interface. At the top, there are three tabs: 'assign\_receptionist.sql', 'select \* from Booking natu ...', and 'select em\_id from ( select ...'. Below these tabs, there are three buttons: 'SQL', 'Output', and 'Statistics'. The 'Output' tab is selected. In the main area, the SQL command 'select \* from Booking natural join Receptionist;' is shown. Below the command, the results of the query are displayed in a grid. The grid has columns: EM\_ID, GUEST\_ID, ROOM\_NUMBER, ENTRY\_DATE, DAYS, SHIFT, LANG. The data consists of 10 rows:

	EM_ID	GUEST_ID	ROOM_NUMBER	ENTRY_DATE	DAYS	SHIFT	LANG	
696	691502586	843604069		31 26/04/1998	...	11	Night	Swahili
697	313776915	999613983		234 20/11/1971	...	5	Evening	Maltese
698	825541986	153296371		496 11/05/2003	...	11	Evening	Assamese
699	410105143	852814667		312 29/04/2012	...	10	Morning	Malayalam
700	147527730	44304219		344 22/10/1989	...	13	Morning	Azeri
701	254183134	301650658		353 25/06/2012	...	8	Morning	Chinese
702	141133895	997645211		560 09/11/1978	...	5	Evening	Bulgarian
703	788436504	578076200		585 13/11/2001	...	10	Morning	Thai
704	433914048	954302901		591 12/01/2010	...	13	Night	Hungarian
705	750280357	983801413		595 12/12/2018	...	8	Evening	Khmer
706	204637922	788085938		567 12/03/1991	...	5	Morning	Portuguese
707	1104267224	242072211		470 05/07/2000	...	8	Morning	Dutch

נ裏ץ את התוכנית הבאה לדוגמה:

```

SQL Output Statistics
begin
    assign_receptionist(954302901, 13, 'English');
end;

```

הפלט שמתתקבל הוא:

```

SQL Output Statistics
Clear Buffer size 10000 Enabled
receptionist 867142561 moved to Morning shift.
guest 954302901 assigned to receptionist 867142561

```

בסיס הנתונים לאחר הריצת התוכנית:

The screenshot shows the results of a query on the Booking table, which includes columns: EM\_ID, GUEST\_ID, ROOM\_NUMBER, ENTRY\_DATE, DAYS, SHIFT, LANG. The data shows various guest entries with their corresponding room numbers, entry dates, stay durations, shifts, and languages.

	EM_ID	GUEST_ID	ROOM_NUMBER	ENTRY_DATE	DAYS	SHIFT	LANG	
699	410105143	852814667		312 29/04/2012	...	10	Morning	Malayalam
700	147527730	44304219		344 22/10/1989	...	13	Morning	Azeri
701	254183134	301650658		353 25/06/2012	...	8	Morning	Chinese
702	141133895	997645211		560 09/11/1978	...	5	Evening	Bulgarian
703	788436504	578076200		585 13/11/2001	...	10	Morning	Thai
704	867142561	954302901		591 12/01/2010	...	13	Morning	English
705	750280357	983801413		595 12/12/2018	...	8	Evening	Khmer
706	204637922	788085938		567 12/03/1991	...	5	Morning	Portuguese
707	118426724	342872311		478 05/07/2002	...	8	Morning	Portuguese
708	427970187	928379443		649 02/06/1971	...	1	Evening	Tsonga
709	750280357	183530934		175 10/01/2007	...	14	Evening	Khmer
710	363052948	795871548		312 12/02/1979	...	10	Night	French
744	740262227	301650657		520 20/11/2000	...	7	Evening	Arabic

### פרוצדורה שנייה:

מבחינת רון, העובדים הם חלק חשוב מההצלחה של בית המלון. רון ידע שכדי לשמר את המוטיבציה והמסירות של העובדים, הוא צריך לתגמל אותם בהתאם. הוא החליט להעניק בונוסים לעובדי המלון שמשכורותם נמוכה מ-10,000 שקלים ושיובילו דירוגים גבוהים על ביצועיהם.

Ron פנה לדן, וביקש ממנו ליצור פרוצדורה שתבצע עדכון אוטומטי למשכורות של העובדים המתאימים. Dan כתוב פרוצדורה בשם `update_salaries_with_bonus` שמבצעת את המשימה זו.

### הקוד ליצירת הפרוצדורה:

```
create or replace procedure update_salaries_with_bonus
```

```

as
cursor low_salary_emp is
  select em_id, salary, rating
  from Employee natural join ReservationAgent
  where salary < 10000 and rating >= 8
  for update of salary;

first_bonus number := 300;
second_bonus number := 200;
third_bonus number := 100;

v_bonus number;

begin
  -- iterate through low salary employees
  for emp in low_salary_emp loop
    -- determine bonus based on salary and rating
    if emp.salary < 8500 and emp.rating >= 9 then
      v_bonus := second_bonus;
    elsif emp.rating = 10 then
      v_bonus := first_bonus;
    else
      v_bonus := third_bonus;
    end if;

    -- update employee salary with bonus
    update employee
    set salary = salary + v_bonus
    where em_id = emp.em_id;
  end loop;

  commit;

exception
  when others then
    dbms_output.put_line('error in update_salaries_with_bonus: ' ||
sqlerrm);
    rollback;
end;

```

לפני הרצת הפרוצדורה:

The screenshot shows the Oracle SQL Developer interface with a query window open. The query is:

```
select *
from Employee natural join ReservationAgent;
```

The results show 18 rows of data from the natural join of the two tables. The columns are: EM\_ID, FIRST\_NAME, LAST\_NAME, SALARY, TECH\_PROFICIENCY, and RATING.

EM_ID	FIRST_NAME	LAST_NAME	SALARY	TECH_PROFICIENCY	RATING
7	Ahmed	Bemlott	7261	Expert	6
8	Umberto	Moreton	7498	Intermediate	9
9	Zacherie	Adan	16869	Expert	7
10	Greg	Oak	7463	Expert	9
11	Margette	Outhwaite	18067	Beginner	4
12	Cherish	Lucok	15470	Intermediate	5
13	Clare	O' Culligan	8000	Beginner	7
14	Rosalyn	Beckson	12273	Expert	4
15	Melodee	McCarrison	8724	Beginner	7
16	Rupert	Balm	16867	Beginner	5
17	Harper	Mace	11734	Expert	3
18	Inabara	Goratt	18517	Beginner	5

נ裏ץ את התוכנית הבאה לדוגמה:

The screenshot shows the Oracle SQL Developer interface with a code editor window open. The code is:

```
begin
    update_salaries_with_bonus;
end;
```

ביסס הנתונים לאחר הריצת התוכנית:

The screenshot shows the Oracle SQL Developer interface with a query window open. The query is:

```
select *
from Employee natural join ReservationAgent;
```

The results show 18 rows of data from the natural join of the two tables. The columns are: EM\_ID, FIRST\_NAME, LAST\_NAME, SALARY, TECH\_PROFICIENCY, and RATING.

EM_ID	FIRST_NAME	LAST_NAME	SALARY	TECH_PROFICIENCY	RATING
7	Ahmed	Bemlott	7261	Expert	6
8	Umberto	Moreton	7698	Intermediate	9
9	Zacherie	Adan	16869	Expert	7
10	Greg	Oak	7663	Expert	9
11	Margette	Outhwaite	18067	Beginner	4
12	Cherish	Lucok	15470	Intermediate	5
13	Clare	O' Culligan	8000	Beginner	7
14	Rosalyn	Beckson	12273	Expert	4
15	Melodee	McCarrison	8724	Beginner	7
16	Rupert	Balm	16867	Beginner	5
17	Harper	Mace	11734	Expert	3

## תוכניות ראשיות:

### תוכנית ראשונה:

התוכנית הבאה משתמשת בפונקציה `find_most_empty_floor` ובפרוצדורה `assign_receptionist`. כיוון שבטיואר של אחת מהן הצגנו הריצה תקינה שלhn עם כל ההוכחות הנדרשות, כאן נריץ עבורה שתיהן דוגמאות שיכשילו אותנו.

עבור הפונקציה: נבחר קבוצה עם תאריך כניסה 01-01-1970 ו-100000 ימי שהות. במצב זה כל החדרים תפוסים ולא נוכל להכניס את הקבוצה בכלל. הפונקציה תדפיס שגיאות.

עבור הפרוצדורה: כניסה ונכנס לבית המלון אורח מסויים בשעה מסוימת שאף מארח לא דבר את שפטו, הפונקציה תחזיר שגיאה.

### קוד התוכנית:

```

declare
    guest_id number;
    check_in_time number;
    guest_lang varchar2(30);
    entry_date date;
    days number;

begin
    -- Using assign_receptionist procedure
    guest_id := '&Enter_your_id';
    check_in_time := '&Enter_your_check_in_hour';
    guest_lang := '&Enter_your_language';
    assign_receptionist(guest_id, check_in_time, guest_lang);

    -- Using find_most_empty_floor function
    entry_date := to_date('&Enter_your_entry_date', 'YYYY-MM-DD');
    days := '&Enter_your_days';
    dbms_output.put_line(find_most_empty_floor(entry_date, days));
end;

```

```

declare
    guest_id number;
    check_in_time number;
    guest_lang varchar2(30);
    entry_date date;
    days number(10);

begin
    -- Using assign_receptionist
    guest_id := '&Enter_your_id';
    check_in_time := '&Enter_your_check_in_hour';
    guest_lang := '&Enter_your_language';
    assign_receptionist(guest_id);

    -- Using find_most_empty_floor
    entry_date := to_date('&Enter_your_entry_date');
    days := '&Enter_your_days';
    dbms_output.put_line(find_most_empty_floor(entry_date, days));
end;

```

הפלט שמתתקבל הוא:

Clear Buffer size 10000 Enabled

an error occurred: ORA-01403: no data found  
ORA-01403: no data found  
Error in find\_most\_empty\_floor: ORA-20001: No empty floor found

שתי שגיאות ראשונות בגל הпроцדורה, ושגיאה שלישית בגל הפונקציה.

### תוכנית שכינית:

התוכנית הבאה משתמשת בפונקציה bed\_average\_price ובprocudr update\_salaries\_with\_bonus. בשתי השאלות האלו לא נוכל לקבל לשגיאה. הפונקציה לא תיכשל כיון שאנו מניחים שהנתונים בטבלאות שלמים, אז תמיד יחזיר ערך כלשהו. לגבי הprocudr, היא תשנה משהו רק במידה ... אם אין מה לשנות היא לא תעשה כלום, אז אין חשש לשגיאה.

למרות שכבר הרצינו דוגמה עברו כל אחת משתיהן והוכחנו שהן עובדות בצורה תקינה, נרץ שוב.

### קוד התוכנית:

```

declare
    v_avg_price number;

begin
    v_avg_price := bed_average_price;
    if v_avg_price is not null then

```

```

        dbms_output.put_line('The average price of a bed in the hotel
is: ' || v_avg_price || '$');
end if;
update_salaries_with_bonus;

end;

```

לא נרצה שינוים בסיס הנתונים אז לא נבצע commit לשינויים. הפלט שקיבلونו:



The screenshot shows the Oracle SQL Developer interface. In the top navigation bar, there are three tabs: 'bed\_average\_price.sql', 'update\_salaries\_with\_bonus.sql', and 'bed\_average\_update\_salaries\_prog.sql'. The 'bed\_average\_update\_salaries\_prog.sql' tab is active. Below the tabs, there are three buttons: 'SQL', 'Output' (which is selected), and 'Statistics'. Underneath these buttons are three controls: 'Clear', 'Buffer size 10000', and a checked checkbox labeled 'Enabled'. The main content area of the window contains the output of the PL/SQL block, which is the message 'The average price of a bed in the hotel is: 205.28\$'.

# אינטגרציה עם אגף ניהול עובדים

הנדסה לאחר של בסיס הנתונים של אגף ניהול עובדים:

זהוקובץsql שקיבלנו מהקבוצה השנייה:

```
CREATE TABLE Shift (
    StartTime      TIMESTAMP NOT NULL,
    EndTime        TIMESTAMP NOT NULL,
    SpecialShift   VARCHAR2(35),
CONSTRAINT pk_Shift PRIMARY KEY (StartTime),
CONSTRAINT check_time CHECK (StartTime < EndTime))
/

CREATE TABLE Department (
    DepartId       NUMBER(5) NOT NULL,
    DepartName     VARCHAR2(35) NOT NULL,
CONSTRAINT pk_Department PRIMARY KEY (DepartId))
/

CREATE TABLE Position (
    PosId          NUMBER(9) NOT NULL,
    DepartId       NUMBER(5) NOT NULL,
    Salary          DECIMAL(5, 2) NOT NULL,
    Role            VARCHAR2(30) NOT NULL,
CONSTRAINT pk_Position PRIMARY KEY (PosId,DepartId),
CONSTRAINT fk_Position FOREIGN KEY (DepartId)
    REFERENCES Department (DepartId)
    ON DELETE CASCADE)
/

CREATE TABLE Employee (
    Id              NUMBER(9) NOT NULL,
    City            VARCHAR2(35) NOT NULL,
    Address         VARCHAR2(35) NOT NULL,
    Phone           VARCHAR2(35) NOT NULL,
    Email           VARCHAR2(45) NOT NULL,
    FirstName       VARCHAR2(35) NOT NULL,
    LastName         VARCHAR2(35) NOT NULL,
    Gender          VARCHAR2(35) NOT NULL,
    JoinDate        DATE NOT NULL,
    PosId           NUMBER(9) NOT NULL,
    DepartId        NUMBER(5) NOT NULL,
CONSTRAINT pk_Employee PRIMARY KEY (Id),
CONSTRAINT fk_Employee FOREIGN KEY (PosId,DepartId)
    REFERENCES Position (PosId,DepartId)
    ON DELETE CASCADE)
/

CREATE TABLE Manager (
    Id              NUMBER(9) NOT NULL,
    AccessType      VARCHAR2(1) NOT NULL,
CONSTRAINT pk_Manager PRIMARY KEY (Id),
CONSTRAINT fk_Manager FOREIGN KEY (Id)
    REFERENCES Employee (Id))
/
```

```

CREATE TABLE Training (
    EntreyLevel      VARCHAR2(35) NOT NULL,
    TrainingId       NUMBER(9) NOT NULL,
    TrainingName     VARCHAR2(35) NOT NULL,
    TrainerId        NUMBER(9),
    Unique(TrainingName,TrainerId,TrainingId),
    CONSTRAINT pk_Tranning PRIMARY KEY (TrainingId),
    CONSTRAINT fk_Tranning FOREIGN KEY (TrainerId)
        REFERENCES Employee (Id))
/

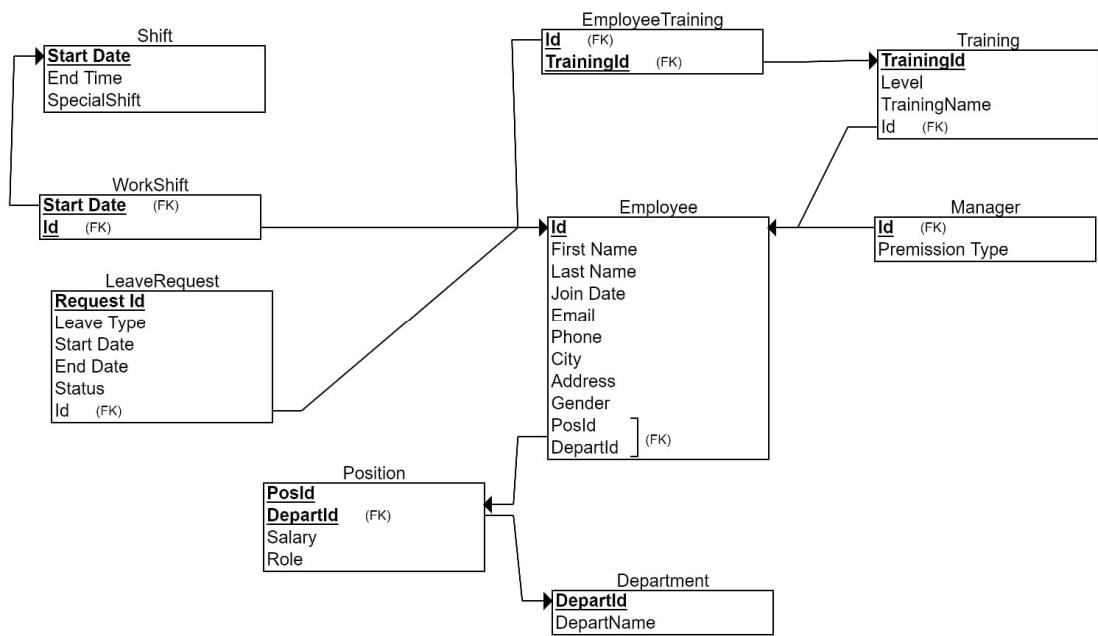
CREATE TABLE LeaveRequest (
    StartDate        DATE NOT NULL,
    EndDate          DATE NOT NULL,
    LeaveType        VARCHAR2(35) NOT NULL,
    RequestId        NUMBER(5) NOT NULL,
    Status           VARCHAR2(35) NOT NULL,
    EmpId            NUMBER(9) NOT NULL,
    CONSTRAINT pk_LeaveRequest PRIMARY KEY (RequestId),
    CONSTRAINT fk_LeaveRequest FOREIGN KEY (EmpId)
        REFERENCES Employee (Id),
    CONSTRAINT check_dates CHECK (StartDate < EndDate))
/

CREATE TABLE EmployeeShift (
    EmpId             NUMBER(9) NOT NULL,
    StartTime         TIMESTAMP NOT NULL,
    CONSTRAINT pk_EmployeeShift PRIMARY KEY (EmpId,StartTime),
    CONSTRAINT fk_EmployeeShift FOREIGN KEY (EmpId)
        REFERENCES Employee (Id)
        ON DELETE CASCADE,
    CONSTRAINT fk_EmployeeShift2 FOREIGN KEY (StartTime)
        REFERENCES Shift (StartTime))
/

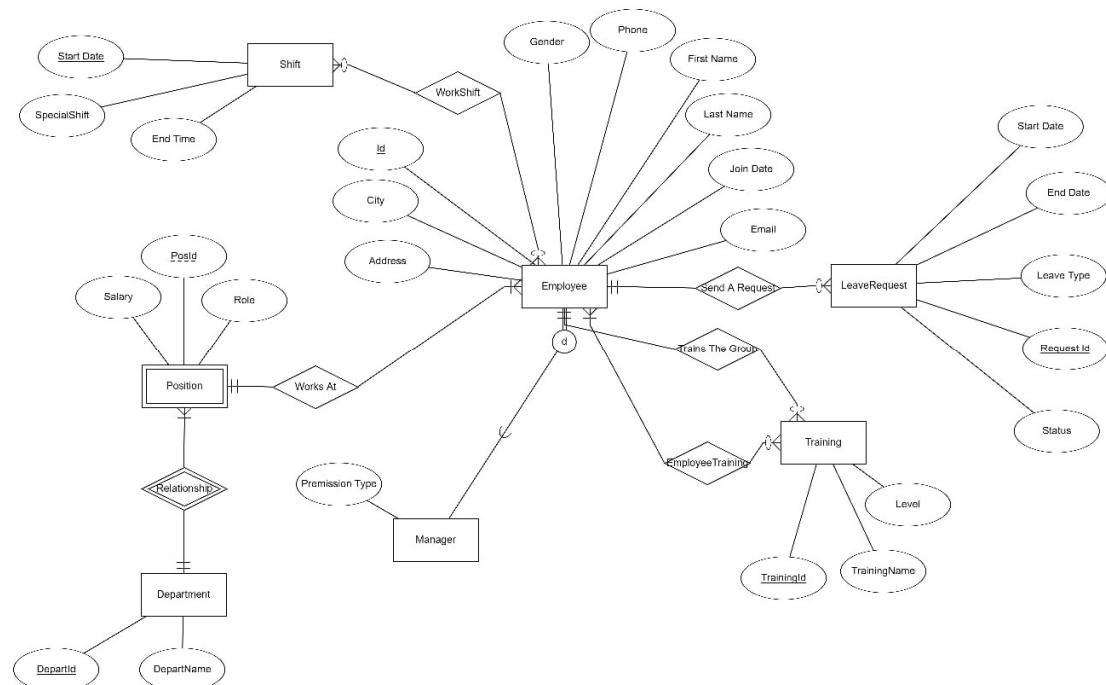
CREATE TABLE EmployeeTraining (
    TrainedId         NUMBER(9) NOT NULL,
    TrainingId        NUMBER(5) NOT NULL,
    CONSTRAINT pk_EmployeeTranning PRIMARY KEY (TrainedId,TrainingId),
    CONSTRAINT fk_EmployeeTranning FOREIGN KEY (TrainedId)
        REFERENCES Employee (Id)
        ON DELETE CASCADE,
    CONSTRAINT fk_EmployeeTranning2 FOREIGN KEY (TrainingId)
        REFERENCES Training (TrainingId))
/

```

יצור ממנה טרשים DSD לפני התוכנות והתלוויות בין היחסות השונות:



נכשוי ביצעו את השלב הבא מעת אחרון בהנדסה לאחר מכן ייצור את תרשימים ה- ERD של האגף לניהול עובדים:



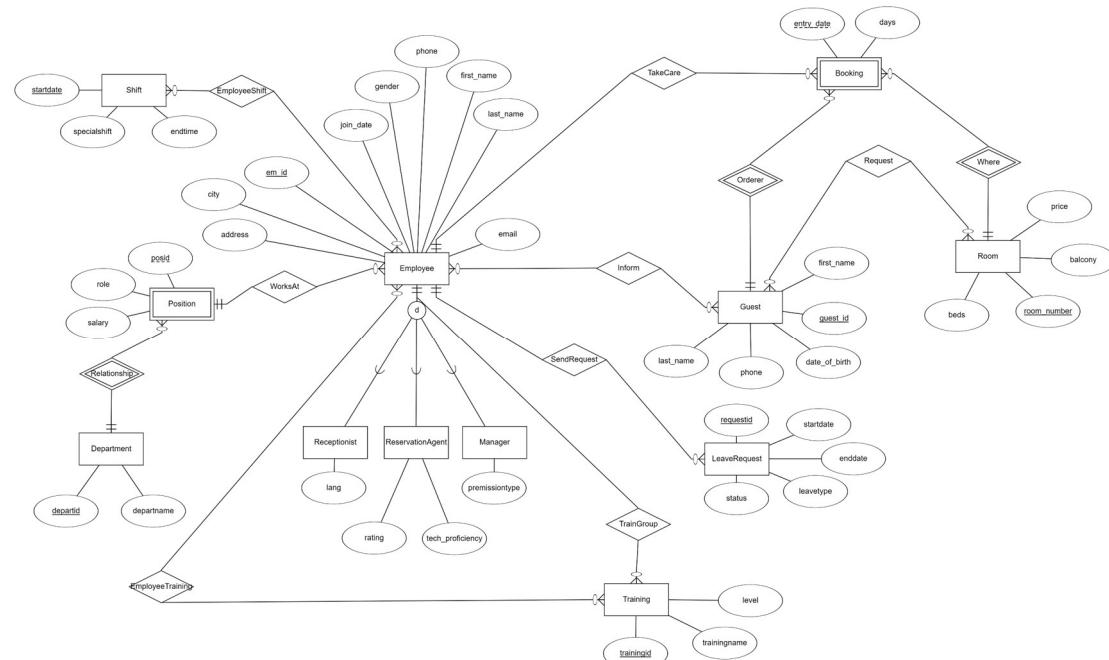
## אינטגרציה ברמת העיצוב:

### תיאור בלוי של השינויים:

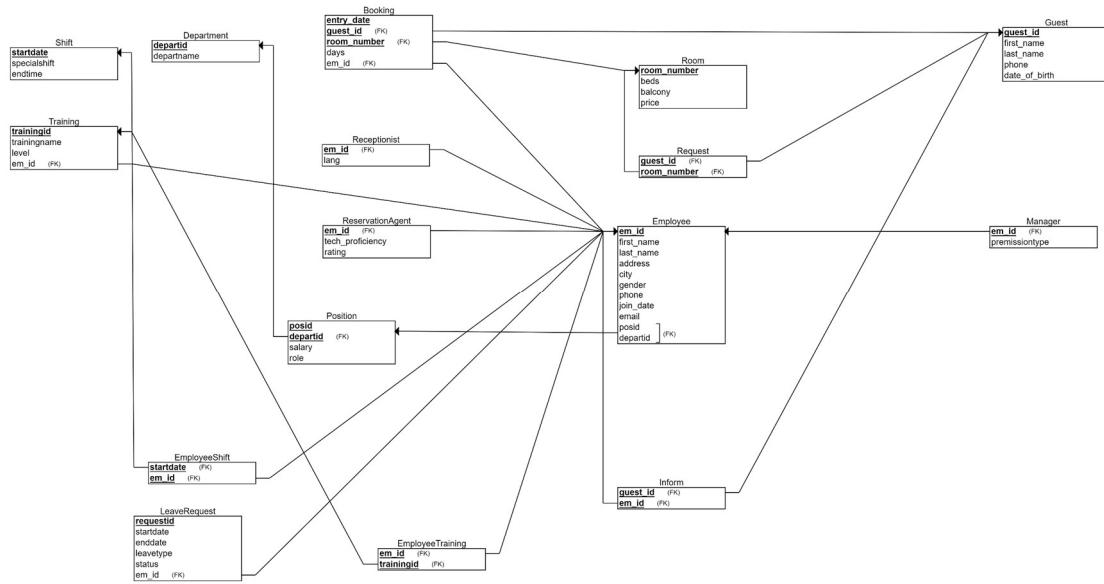
1. נרצה להשתמש ב- Employee שлемו ולהוסיף לו את התכונת והערכים של Employee2. אם ככה, המפתחות id\_em שלמו לא מתאימים לכל הזרים שנמצאים בטבלאות, אז נדרש לעדכן את כל העמודות id לערבי המפתחות שלמו. ההחלפה תהיה בטבלאות EmployeeShift, LeaveRequest, Training, EmployeeTraining, Manager.
2. ביוון שקיבלנו ישות Shift מאגף ניהול עובדים, נוכל לוותר על התכונה shift שבטבלה Position. כמו כן, התכונה salary מופיעה בטבלה Position שקיבלנו, אך נמחק אותה מהטבלה Employee.

בצורה זו ניצור מערכת משולבת שתומכת בכל הfonקציונליות המקורית הן שלמו והן של האגף השני.

### תרשים ERD משולב של אגף ניהול עובדים ואגף ניהול הזמן:



## תרשים DSD משולב של אגף ניהול עובדים ואגף ניהול הזמן:



## אינטגרציה ברמת הנתונים:

לשם הנוחות, כדי שלא תהיה התנגדות, נחליף כל Employee מקובץ הגיבוי שקיבלנו ל-2 Employee (כיו בסוף נמחק את הטבלה). עבשו ניבא את בסיס הנתונים של אגף ניהול עובדים לאזרור בסיס הנתונים שלנו. נשים לב שיש אילוץ UNION בטבלה Training שקיים שם שאין לו שם, אז ניתן לו שם ידנית – `qq_training`.

## תיאור מפורט של הדרך לאינטגרציה מלאה:

נתיחס לכל אחד משני השינויים לעיל בנפרד ונפרט עבור כל אחד, מהו סדר הפעולות הדרוש על מנת להשיג את השינוי (שילוב) המבוקש. המטרה בגודול תהיה לבסס את המערכת המשולבת על הנתונים המקוריים שלנו ואיופה לצריך להוסיף עמודות מתאימות מבוסיס הנתונים של האגף השני.

שינויי ראשוני:

- געתיκ את כל התכונות החסרות מהטבלה Employee2 אל הטבלה Employee.
- מוסיף לשתי הטבלאות Employee2 ו- Employee2 עמודה temp עם ערכי ROWNUM.
- געתיκ את כל הערכים על פי מספר השורה.
- נגדיר את `pi_pos_id`, `depart_id`, `depart_id` בטבלה Position, שמתיחס לעמודות אלו.
- מוסיף עמודה זמנית `pi_temp` לטבלה Employee.

- f. נעתיק את ערכי ה- `po` המקוריים שלהם לעמודה `id_temp` שיצרנו בשלב d.
- g. נמחק את עמודות `temp`. מעבשו כל `po` שMOVED עצם בטבלאות `LeaveRequest, Training, EmployeeTraining, Manager` המתאים שבטבלה `Employee`. השלבים הבאים יבצעו זאת.
- h. נמחק בחמש הטבלאות את התלות בתוכנה `po` המקורי שליהם.
- i. נמחק את ה- `key primary` בכל אחת מחמש הטבלאות (כדי שטобל לעבור לשלב הבא), לשם כך נדרש להסיר את התלות של `EmployeeTraining` ב- `Training`.
- j. נוסיף עמודה `id_em` לכל אחת מחמש הטבלאות שצוינו לעיל.
- k. נעדכן את העמודות `id_em` לערכים המתאימים לעמודה `po`, לפי הטבלה `Employee`.
- l. נמחק את כל עמודות `po` (לשםותיהן השונות) בחמש הטבלאות, וגם את `id_temp` בטבלה `Employee`. לפני כן נדרש להסיר את האילוץ `UNIQUE` מהטבלה `Training`.
- m. נעדכן את כל האילוצים שהסכנו בשלבים הקודמים כדי להחזיר את הטבלאות למבנה הקודם שלון.
- n. נשאיר את הטבלה `Employee2`, אין לנו בה שום צורך.
- o. כדי לשדרג את ההיגיון במערכת המשולבת שלנו, נשנה את תוכנת ה- `role` בטבלה `Position` לפי הבנים של `Employee`. למשל עבור כל עובד מסווג `Receptionist`, נשנה את עמודת ה- `role` ברשותה ה- `Position` המתאימה, ל- '`Receptionist`'.

**שיכוני שני:**

- a. נסיר את העמודה `shift` מהטבלה `Receptionist`.
- b. נסיר את העמודה `salary` מהטבלה `Employee`.

#### הקוד שמבצע את האינטגרציה בין האגפים:

```
----- Step 1 -----
--- a:
alter table Employee
add city varchar2(35) default 'Unknown';

alter table Employee
add address varchar2(35) default 'Unknown';

alter table Employee
add phone varchar2(35) default 'Unknown';

alter table Employee
add email varchar2(45) default 'Unknown';

alter table Employee
add gender varchar2(35) default 'Unknown';

alter table Employee
```

```

add join_date date default date '1970-01-01';

alter table Employee
add pos_id number(9) default 0;

alter table Employee
add depart_id number(5) default 0;

--- b:
alter table Employee
add temp int default 0;

update Employee
set temp = rownum;

alter table Employee2
add temp int default 0;

update Employee2
set temp = rownum;

--- c:
update Employee
set
  city = (
    select Employee2.city
    from Employee2
    where Employee2.temp = mod(Employee.temp - 1, (select
count(*) from Employee2)) + 1
  ),
  address = (
    select Employee2.address
    from Employee2
    where Employee2.temp = mod(Employee.temp - 1, (select
count(*) from Employee2)) + 1
  ),
  phone = (
    select Employee2.phone
    from Employee2
    where Employee2.temp = mod(Employee.temp - 1, (select
count(*) from Employee2)) + 1
  ),
  email = (
    select Employee2.email
    from Employee2
    where Employee2.temp = mod(Employee.temp - 1, (select
count(*) from Employee2)) + 1
  ),
  gender = (
    select Employee2.gender
    from Employee2
    where Employee2.temp = mod(Employee.temp - 1, (select
count(*) from Employee2)) + 1
  ),
  join_date = (
    select Employee2.joindate
    from Employee2
  )

```

```

        where Employee2.temp = mod(Employee.temp - 1, (select
count(*) from Employee2)) + 1
),
pos_id = (
    select Employee2.posid
    from Employee2
    where Employee2.temp = mod(Employee.temp - 1, (select
count(*) from Employee2)) + 1
),
depart_id = (
    select Employee2.departid
    from Employee2
    where Employee2.temp = mod(Employee.temp - 1, (select
count(*) from Employee2)) + 1
);

```

--- d:

```

alter table Employee
add constraint fk_pos_depart_id_employee
foreign key (pos_id, depart_id) references Position (posid, departid)
on delete cascade;

```

--- e:

```

alter table Employee
add temp_id int default null;

```

--- f:

```

update Employee
set
    temp_id = (
        select Employee2.id
        from Employee2
        where Employee2.temp = Employee.temp
    );

```

--- g:

```

alter table Employee
drop column temp;

alter table Employee2
drop column temp;

```

--- h:

```

alter table EmployeeShift
drop constraint fk_employeeshift;

alter table LeaveRequest
drop constraint fk_leaverequest;

alter table Training
drop constraint fk_tranning;

```

```

alter table EmployeeTraining
drop constraint fk_employee2tranning;

alter table Manager
drop constraint fk_manager2;

--- i:
alter table Employeetraining
drop constraint fk_employee2tranning2;

alter table EmployeeShift
drop constraint pk_employeeshift;

alter table LeaveRequest
drop constraint pk_leaverequest;

alter table Training
drop constraint pk_tranning;

alter table EmployeeTraining
drop constraint pk_employee2tranning;

alter table Manager
drop constraint pk_manager;

--- j:
alter table EmployeeShift
add em_id int default 0;

alter table LeaveRequest
add em_id int default 0;

alter table Training
add em_id int default 0;

alter table EmployeeTraining
add em_id int default 0;

alter table Manager
add em_id int default 0;

--- k:
update EmployeeShift
set
    em_id = (
        select Employee.em_id
        from Employee
        where EmployeeShift.empid = Employee.temp_id
    );

update LeaveRequest
set
    em_id = (

```

```

        select Employee.em_id
        from Employee
        where LeaveRequest.empid = Employee.temp_id
    ) ;

update Training
set
    em_id = (
        select Employee.em_id
        from Employee
        where Training.trainerid = Employee.temp_id
    ) ;

update EmployeeTraining
set
    em_id = (
        select Employee.em_id
        from Employee
        where EmployeeTraining.trainedid = Employee.temp_id
    ) ;

update Manager
set
    em_id = (
        select Employee.em_id
        from Employee
        where Manager.id = Employee.temp_id
    ) ;

--- l:
alter table Training
drop constraint uq_training;

alter table EmployeeShift
drop column empid;

alter table LeaveRequest
drop column empid;

alter table Training
drop column trainerid;

alter table EmployeeTraining
drop column trainedid;

alter table Manager
drop column id;

--- m:
alter table EmployeeShift
add constraint PK_EMPLOYEESHIFT
primary key (em_id, starttime);

alter table LeaveRequest
add constraint PK_LEAVEREQUEST
primary key (requestid);

```

```

alter table Training
add constraint PK_TRAINING
primary key (trainingid);

alter table EmployeeTraining
add constraint PK_EMPLOYEE2TRAINING
primary key (em_id, trainingid);

alter table Manager
add constraint pk_manager
primary key (em_id);

-----
alter table EmployeeTraining
add constraint FK_EMPLOYEE2TRAINING2
foreign key (trainingid) references Training (trainingid) on delete cascade;

alter table EmployeeShift
add constraint FK_EMPLOYEESHIFT
foreign key (em_id) references Employee (em_id) on delete cascade;

alter table LeaveRequest
add constraint FK_LEAVEREQUEST
foreign key (em_id) references Employee (em_id) on delete cascade;

alter table Training
add constraint FK_TRAINING
foreign key (em_id) references Employee (em_id) on delete cascade;

alter table EmployeeTraining
add constraint FK_EMPLOYEE2TRAINING
foreign key (em_id) references Employee (em_id) on delete cascade;

alter table Manager
add constraint FK_MANAGER2
foreign key (em_id) references Employee (em_id) on delete cascade;

--- n:
drop table Employee2;

--- o:
update Position
set role = 'Receptionist'
where exists (
    select *
    from Employee e
    natural join Receptionist r
    where e.pos_id = Position.posid
    and e.depart_id = Position.departid
);

update Position
set role = 'Reservation Agent'

```

```
where exists (
    select *
    from Employee e
    natural join ReservationAgent ra
    where e.pos_id = Position.posid
    and e.depart_id = Position.departid
);
```

----- Step 2 -----

```
--- a:
alter table Receptionist
drop column shift;

--- b:
alter table Employee
drop column salary;
```

ניתן לראות בקלות איזה קטע מטור הקוד מתאים לכל אחד מהשלבים המפורטים לעיל.

# מבטיים

## הגדרת ה- Views:

### View ראשון:

אגף ניהול העובדים של בית המלון מעוניין באפשרות להציג ארכו על נתונים העובדים במערכת בית המלון. לא מעוניינים אותו האורחים וכל דברים האחרים בעסקי בית המלון. נכתב עבורה view שיביל רק את הנתונים על העובדים.

### הקוד ליצירת ה- view:

```
--- View Definition:  
create or replace view EmployeeDetailsView as  
select  
    e.em_id,  
    e.first_name,  
    e.last_name,  
    e.email,  
    e.phone,  
    d.departName,  
    p.role,  
    p.salary,  
    m.accessType as managerAccessType  
from  
    Employee e  
    join Position p on e.pos_id = p.posId and e.depart_id =  
p.departId  
    join Department d on e.depart_id = d.departId  
    left join Manager m on e.em_id = m.em_id;
```

### שליפה נתונים מתוך ה- view:

The screenshot shows the SQL Server Management Studio interface with a results grid displaying employee details. The columns are: EM\_ID, FIRST\_NAME, LAST\_NAME, EMAIL, PHONE, DEPARTNAME, ROLE, SALARY, and MANAGERACCESSTYPE. The data includes 12 rows of employee information such as Lilian Beardsdale, Elsa Bewshea, Alyssa McCuish, Stephen Volkes, etc., with their respective department names, roles, salaries, and manager access types.

	EM_ID	FIRST_NAME	LAST_NAME	EMAIL	PHONE	DEPARTNAME	ROLE	SALARY	MANAGERACCESSTYPE
1	867142561	Lilian	Beardsdale	edlink2@hud.gov	(751) 5853878	Human Resources	Reservation Agent	44.35	
2	750280357	Elsa	Bewshea	cangelo5@tund1.de	(906) 444269	Human Resources	Receptionist	41.23	
3	980508742	Alyssa	McCuish	slovekinc@columbia.edu	(400) 6932574	Human Resources	Recruiter	41.23	
4	601965364	Stephen	Volkes	darmitt@tinyurl.com	(693) 4266291	Human Resources	Recruiter	41.23	
5	236653050	Millisent	Hearons	chedgerj@cdc.gov	(258) 3665588	Human Resources	Receptionist	41.23	
6	750114742	Toddy	Pinner	lstenhousen@goo.gl	(304) 9954129	Human Resources	Recruiter	41.23	
7	195231112	Salaidh	Blackford	zwillingleq@mariott.com	(634) 6196978	Human Resources	Recruiter	41.23	
8	45087593	Chloris	Sutherns	ohavickt@google.ca	(103) 9864707	Human Resources	Reservation Agent	41.23	
9	815386042	Estelle	Hise	cbloano@hhs.gov	(710) 7503416	Human Resources	Receptionist	41.23	
10	50555545	Meredes	Aish	mdickin10@mail.ru	(422) 2309054	Human Resources	Recruiter	41.23	
11	736009235	Estrellita	Grichukhanov	ecristofaro14@privacy.gov.au	(242) 8734905	Human Resources	Receptionist	41.23	
12	204637922	Erik	Ianelli	ddouthch17@nydailynews.com	(632) 9042179	Human Resources	Receptionist	41.23	

## View שבין

างף ניהול האורחים וההזמנות מעוניין לאסוף את נתוני הפעולות של המארחים ושל סוכני ההזמנות השונים. על כל עובד נרצה לדעת מהו מספר האורחים להם העניק שירות, כאשר השירות שמעניק מארח הוא טיפול ודאגה לאורח, והשירות שמעניק סוכן ההזמנות הוא תמייבה לפני שלב ההזמנה עצמו. ניצור view עבור האגף לפי הדרישת שלו.

## הקוד ליצירת ה-view:

```
--- View Definition:
create or replace view HotelStaffActivityView as
select
    e.em_id,
    e.first_name as employee_first_name,
    e.last_name as employee_last_name,
    case
        when ra.em_id is not null then 'Reservation Agent'
        when r.em_id is not null then 'Receptionist'
    end as employee_type,
    count(distinct b.guest_id) as total_bookings_handled,
    count(distinct i.guest_id) as total_guests_informed
from
    Employee e
    inner join (
        select em_id, 'Reservation Agent' as type from ReservationAgent
        union
        select em_id, 'Receptionist' as type from Receptionist
    ) staff on e.em_id = staff.em_id
    left join ReservationAgent ra on e.em_id = ra.em_id
    left join Receptionist r on e.em_id = r.em_id
    left join Booking b on e.em_id = b.em_id
    left join Inform i on e.em_id = i.em_id
group by
    e.em_id, e.first_name, e.last_name,
    case
        when ra.em_id is not null then 'Reservation Agent'
        when r.em_id is not null then 'Receptionist'
    end
order by
    total_bookings_handled desc, total_guests_informed desc;
```

## שליפת נתונים מתוך ה-view:

The screenshot shows a SQL Server Management Studio window with the following details:

- Tab bar: createTables.sql, dropTables.sql, Integrate.sql, Query data C##SQL.HOTELSTAFFACTIVITYVIEW@XE
- Toolbar: SQL, Output, Statistics
- Query pane: select \* from HOTELSTAFFACTIVITYVIEW t
- Results pane: A grid displaying 12 rows of data from the view.

**View Data:**

EM_ID	EMPLOYEE_FIRST_NAME	EMPLOYEE_LAST_NAME	EMPLOYEE_TYPE	TOTAL_BOOKINGS_HANDLED	TOTAL_GUESTS_INFORMED
1	196295696	Cathrin	Receptionist	12	0
2	216049651	Lorin	Receptionist	11	0
3	895641532	Odey	Receptionist	10	0
4	518947468	Fawn	Receptionist	9	0
5	427970187	Jerilyn	Receptionist	9	0
6	262380974	Tierney	Receptionist	9	0
7	602194268	Norri	Receptionist	9	0
8	944171827	Cory	Receptionist	8	0
9	718408998	Brigid	Receptionist	8	0
10	788436504	Leopold	Receptionist	8	0
11	629672609	Mischa	Receptionist	8	0
12	254183134	Athena	Receptionist	7	0

## שאילות על ה-Views:

### שאילה על ה-View הראשון:

רואה החשבון הוותיק של בית המלון, יוסי, יצא לפנסיה עוד בשבועיים. גדי, רואה החשבון הטרוי, החליף אותו והם בעת בשלב החיפויה. גדי מעוניין לבדוק כמה עובדים מקבלים משכורת שעהית של מעל נס. 70.

*--- Query for Example:*

```
select
    first_name,
    last_name,
    departName,
    role,
    salary
from
    EmployeeDetailsView
where
    salary > 70
order by
    salary desc;
```

פלט השאלתה:

```

SQL Output Statistics
--- Query for Example:
select
    first_name,
    last_name,
    departName,
    role,
    ``
Create employeeDetailsView Select employeeDetailsView
| FIRST_NAME | LAST_NAME | DEPARTMENTNAME | ROLE | SALARY |
| --- | --- | --- | --- | --- |
| 1 Taffy | " Gaunson | " Human Resources | Receptionist | 110.77 |
| 2 Daffi | " Douglass | " Human Resources | Receptionist | 110.77 |
| 3 Shaw | " Blanchette | " Sales | Receptionist | 100.70 |
| 4 Francisco | " Shovelin | " Security | Receptionist | 100.70 |
| 5 Jeremiah | " Tricker | " IT & Tech | Reservation Agent | 100.70 |
| 6 Jeth | " Wallcker | " Restaurant | Reservation Agent | 100.70 |
| 7 Constanca | " Norvill | " Finance | Manager | 100.70 |
| 8 Catherine | " Jenteau | " Housekeeping | Manager | 100.70 |
| 9 Patrizio | " Mayhew | " Restaurant | Chef | 88.46 |
| 10 Marnie | " Kleinstern | " Restaurant | Chef | 88.46 |

```

### שאילתת על ה- View השניה:

לשכת הסטטיסטיקה של בית המלון מעוניינת לבדוק נתונים כללים עבור כל המארחים וכן עבור כל סוכני ההזמנות. עבור קבוצת המארחים נרצה לבדוק: 1) מהו מספר ההזמנות הממוצע בהן מטפל כל אחד, 2) מהו מספר ההזמנות הכולל שוטוף על ידי המארחים של בית המלון. עבור קבוצת סוכני ההזמנות נרצה לבדוק: 1) מהו מספר האורחים הממוצע איתם כל סוכן הזמנות מתקשר, 2) מהו מספר הקשרים הכולל שנוצר בין המארחים לסוכני ההזמנות.

```

--- Query for Example:
select
    employee_type,
    count(*) as staff_count,
    avg(total_bookings_handled) as avg_bookings_per_employee,
    avg(total_guests_informed) as avg_guests_informed_per_employee,
    sum(total_bookings_handled) as total_bookings,
    sum(total_guests_informed) as total_guests_informed
from
    HotelStaffActivityView
group by
    employee_type
order by
    avg_bookings_per_employee desc;

```

### פלט השאלה:

```

SQL Output Statistics
--- Query for Example:
select
    employee_type,
    count(*) as staff_count,
    avg(total_bookings_handled) as avg_bookings_per_employee,
    avg(total_guests_informed) as avg_guests_informed_per_employee,
    sum(total_bookings_handled) as total_bookings,
    sum(total_guests_informed) as total_guests_informed
from
    HotelStaffActivityView
group by
    employee_type
Create hotelstaffactivityview Select hotelstaffactivityview
| EMPLOYEE_TYPE | STAFF_COUNT | AVG_BOOKINGS_PER_EMPLOYEE | AVG_GUESTS_INFORMED_PER_EMPLOYEE | TOTAL_BOOKINGS | TOTAL_GUESTS_INFORMED |
| --- | --- | --- | --- | --- | --- |
| 1 Receptionist | 394 | 3.79695431472081 | 0 | 1496 | 0 |
| 2 Reservation Agent | 252 | 0 | 1.19047619047619 | 0 | 300 |

```