

מיני פרויקט בבסיסי נתונים

רועי סאיג 327839858 - הראל ברנר

מערכת לניהול בתי מלון

לוגיסטיקה והזמנות ציוד

תוכן עניינים:

דו"ח 1 - 3-16

דו"ח 2 - 17-29

דו"ח 3 - 30-40

דו"ח 4 - 41-60

דו"ח 1 - מיני פרויקט בבסיסי נתונים
רועי סאיג 327839858 - הראל ברנר
מערכת לניהול בתי מלון
לוגיסטיקה והזמנות ציוד

תיאור מילולי:

מערכת לניהול תקלות בבית מלון נועדה לטפל ביעילות בתקלות ובבקשות תחזוקה שונות המתעוררות בבית המלון. המערכת כוללת שישה מרכיבים מרכזיים, שכל אחד מהם ממלא תפקיד חשוב בתהליך הניהול והתחזוקה. להלן תיאור מפורט של כל אחד מהמרכיבים:

1. מחלקה (Department):

- תיאור: מחלקות שונות בבית המלון האחראיות על סוגי שירותים ותחזוקה שונים. למשל: חשמל, אינסטלציה, נגרות, ניקיון ועוד.

- מטרת המחלקה: לנהל את התחזוקה והשירותים בתחומה ולוודא שכל תקלה מטופלת במהירות וביעילות.

2. עובד (Employee):

- תיאור: עובדים המועסקים במחלקות השונות של בית המלון. כל עובד משתייך למחלקה מסוימת ויש לו כישורים המתאימים לתחום אחריותו.

- מטרת העובד: לטפל בבקשות תחזוקה ותקלות בתחום אחריותו.

3. מיקום (Location):

- תיאור: האזורים השונים בבית המלון שבהם יכולה להתרחש תקלה או שנדרש בהם שירות תחזוקה. למשל: חדרים, לובי, מטבחים, חדרי ישיבות ועוד.

- מטרת המיקום: לאפשר ניהול ותיעוד מדויק של מיקום התקלה או הבקשה.

4. ציוד (Equipment):

- תיאור: הציוד והכלים בהם משתמשים העובדים במחלקות השונות לביצוע עבודות תחזוקה ושירותים. למשל: כלי עבודה, מכשירי חשמל, חומרי ניקוי ועוד.

- מטרת הציוד: לספק לעובדים את הכלים הנדרשים לביצוע המשימות שלהם.

5. בקשת תחזוקה (Maintenance Request):

- תיאור: בקשות תחזוקה המוגשות על ידי אורחי המלון או הצוות לזיהוי תקלות או צורך בתחזוקה במיקום מסוים בבית המלון.

- מטרת בקשת התחזוקה: לנהל ולעקוב אחר תקלות ובקשות שירות כדי לוודא שהן מטופלות בזמן ובצורה יעילה.

6. משימה (Task):

- תיאור: משימות ספציפיות שמוקצות לעובדים במחלקות השונות בהתאם לבקשות תחזוקה שהוגשו.

- מטרת המשימה: להבטיח שכל בקשת תחזוקה תטופל על ידי העובד המתאים בזמן המתאים. יתרונות המערכת:

- ניהול יעיל: המערכת מאפשרת ניהול יעיל ומהיר של תקלות ובקשות תחזוקה.

- שיפור השירות: טיפול מהיר בתקלות משפר את חווית האורחים בבית המלון.

- מעקב ותיעוד: המערכת מאפשרת מעקב ותיעוד של כל התקלות והמשימות שבוצעו, מה שמסייע בשיפור תהליכים עתידיים.

- ניהול משאבים: המערכת מסייעת בניהול טוב יותר של העובדים והציוד, כך שניתן למקסם את היעילות.

באמצעות מערכת זו, בית המלון יכול להבטיח שכל תקלה תטופל בצורה מהירה ומקצועית, תוך ניהול יעיל של המשאבים והעובדים.

ישויות:

1. חדר
2. בקשת תחזוקה
3. עובד
4. ציוד
5. משימה
6. מחלקה

פירוט תפקידים:

חדר (Location):

מתאר את מיקום התקלה בבית המלון.

לכל מיקום יכולים להיות מספר בקשות תחזוקה.

בקשת תחזוקה (MaintenanceRequest):

מתאר את בקשות התחזוקה השונות שיש בחדרי המלון או בלובי ובמסדרונות ומחלק אותם לפני מחלקות ודחיפות.

כל בקשת תחזוקה מתייחסת למיקום אחד, למשימה אחת ולמחלקה אחת בלבד.

עובד (Employee):

מתאר עובד בבית המלון המשוך למחלקה מסוימת (תחום עיסוק).

כל עובד מיוחס רק למחלקה אחת.

ציוד (Equipment):

מתאר את כלי העבודה שבהם משתמשים העובדים.

כל מכשיר מיוחס למחלקה אחת בלבד.

משימה (Task):

מתאר משימה לטיפול בבקשת תחזוקה.

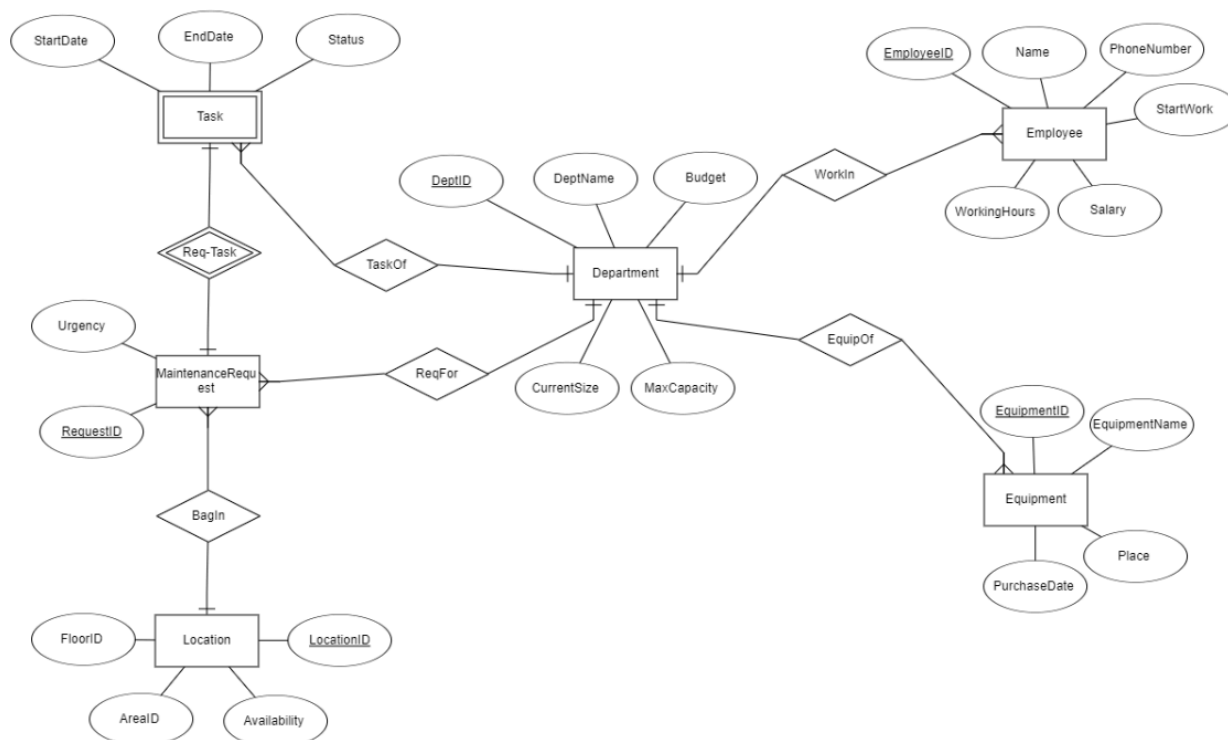
כל משימה מתייחסת לבקשת תחזוקה אחת ולמחלקה אחת בלבד.

מחלקה (Department):

מתאר את חלוקת העבודה לפי תחומי עיסוק (כמו חשמל, אינסטלציה, ניקיון וכו')

לכל מחלקה יכולים להיות כמה עובדים, מכשירים, בקשות תחזוקה ומשימות.

תרשים ERD:



פירוט תכונות:

מיקום (Location):

מיקום(מפתח), מספר קומה, איזור, זמינות(פנוי/תפוס)

בקשת תחזוקה (MaintenanceRequest):

מספר בקשה(מפתח), מיקום(מפתח זר), מספר מחלקה(מפתח זר), דחיפות

עובד (Employee):

מספר עובד (מפתח), שם עובד, מספר טלפון, תאריך תחילת עבודה, מס מחלקה (מפתח זר), משכורת, מספר שעות עבודה.

ציוד (Equipment):

מספר מכשיר (מפתח), שם מכשיר, מספר מחלקה (מפתח זר), מיקום המכשיר, תאריך רכישה.

משימה (Task):

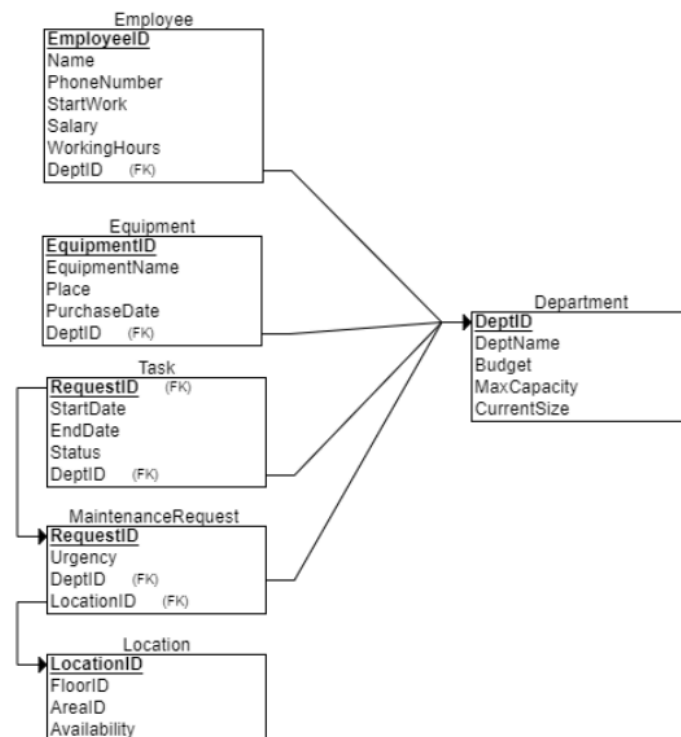
מספר בקשה (מפתח + מפתח זר), מספר מחלקה (מפתח זר), תאריך התחלת המשימה, תאריך סיום משוער, מצב (טופל או לא).

מחלקה (Department):

מספר מחלקה (מפתח), שם מחלקה, תקציב, גודל מקסימלי למחלקה, גודל בפועל.

- Location_ (LocationID, FloorID, AreaID, Availability)
- MaintenanceRequest (RequestID, LocationID(FK), DeptID(FK), Urgency)
- Employee (EmployeeID, Name, PhoneNumber, StartWork, DeptID(FK), Salary, WorkingHours)
- Equipment (EquipmentID, EquipmentName, DeptID(FK), Place, PurchaseDate)
- Task (RequestID(FK+PK), DeptID(FK), StartDate, EndDate, Status)
- Department (DeptID, DeptName, Budget, MaxCapacity, CurrentSize)

תרשים DSD:



פקודות create table:

Department:

```

CREATE TABLE Department (
    DeptID INT,
    DeptName VARCHAR(50) NOT NULL,
    Budget INT NOT NULL,
    MaxCapacity INT NOT NULL,
    CurrentSize INT NOT NULL,
    CONSTRAINT PK_Department PRIMARY KEY (DeptID)
);
    
```


:Employee7

```

CREATE TABLE Employee (
    EmployeeID INT,
    Name VARCHAR(50) NOT NULL,
    PhoneNumber VARCHAR(20) NOT NULL,
    CONSTRAINT Phone_Format CHECK (
        REGEXP_LIKE(PhoneNumber, '^\d{3}-\d{3}-\d{4}$')),

    StartWork DATE NOT NULL,
    Salary INT NOT NULL,
    WorkingHours INT NOT NULL,
    DeptID INT NOT NULL,
    CONSTRAINT PK_Employee PRIMARY KEY (EmployeeID),
    CONSTRAINT FK_Employee_Dept FOREIGN KEY (DeptID) REFERENCES Department(DeptID)
);

```

Equipment7

```

CREATE TABLE Equipment (
    EquipmentID INT,
    EquipmentName VARCHAR(50) NOT NULL,
    Place VARCHAR(50) NOT NULL,
    PurchaseDate DATE NOT NULL,
    DeptID INT NOT NULL,
    CONSTRAINT PK_Equipment PRIMARY KEY (EquipmentID),
    CONSTRAINT FK_Equipment_Dept FOREIGN KEY (DeptID) REFERENCES Department(DeptID)
);

```

:Location7

```

CREATE TABLE Location (
    LocationID INT,
    FloorID INT NOT NULL,
    AreaID VARCHAR(15) NOT NULL,
    Availability VARCHAR(10) NOT NULL,
    CONSTRAINT PK_Location PRIMARY KEY (LocationID)
);

```

:MaintenanceRequest

```
CREATE TABLE MaintenanceRequest (
    RequestID INT,
    Urgency VARCHAR(10) NOT NULL,
    DeptID INT NOT NULL,
    LocationID INT NOT NULL,
    CONSTRAINT PK_MaintenanceRequest PRIMARY KEY (RequestID),
    CONSTRAINT FK_MaintenanceRequest_Dept FOREIGN KEY (DeptID) REFERENCES Department (DeptID),
    CONSTRAINT FK_MaintenanceRequest_Location FOREIGN KEY (LocationID) REFERENCES Location (LocationID)
);
```

Task

```
CREATE TABLE Task (
    RequestID INT,
    StartDate DATE NOT NULL,
    EndDate DATE NOT NULL,
    Status VARCHAR(20) NOT NULL,
    DeptID INT NOT NULL,
    CONSTRAINT PK_Task PRIMARY KEY (RequestID),
    CONSTRAINT FK_Task_MaintenanceRequest FOREIGN KEY (RequestID) REFERENCES MaintenanceRequest_ (RequestID),
    CONSTRAINT FK_Task_Dept FOREIGN KEY (DeptID) REFERENCES Department (DeptID)
);
```

פקודות drop table:

```
-- Drop Location table
DROP TABLE Location;

-- Drop MaintenanceRequest table
DROP TABLE MaintenanceRequest;

-- Drop Employee table
DROP TABLE Employee;

-- Drop Equipment table
DROP TABLE Equipment;

-- Drop Department table
DROP TABLE Department;

-- Drop Task table
DROP TABLE Task;
```

פקודת insert table:

לDepartment:

```
-- Insert into Department
INSERT INTO Department (DeptID, DeptName, Budget, MaxCapacity, CurrentSize) VALUES
(1, 'Engineering', 100000, 50, 30);
INSERT INTO Department (DeptID, DeptName, Budget, MaxCapacity, CurrentSize) VALUES
(2, 'Human Resources', 50000, 10, 7);
INSERT INTO Department (DeptID, DeptName, Budget, MaxCapacity, CurrentSize) VALUES
(3, 'Maintenance', 75000, 20, 15);
```

לEmployee:

```
-- Insert into Employee
INSERT INTO Employee (EmployeeID, Name, PhoneNumber, StartWork, Salary, WorkingHours, DeptID) VALUES
(1, 'John Doe', '123-456-7890', DATE '2020-01-15', 60000, 40, 1);
INSERT INTO Employee (EmployeeID, Name, PhoneNumber, StartWork, Salary, WorkingHours, DeptID) VALUES
(2, 'Jane Smith', '234-567-8901', DATE '2018-06-01', 55000, 40, 2);
INSERT INTO Employee (EmployeeID, Name, PhoneNumber, StartWork, Salary, WorkingHours, DeptID) VALUES
(3, 'Emily Johnson', '345-678-9012', DATE '2019-03-20', 50000, 40, 3);
```

לEquipment:

```
-- Insert into Equipment
INSERT INTO Equipment (EquipmentID, EquipmentName, Place, PurchaseDate, DeptID) VALUES
(1, 'Laptop', 'Office', DATE '2022-02-15', 1);
INSERT INTO Equipment (EquipmentID, EquipmentName, Place, PurchaseDate, DeptID) VALUES
(2, 'Projector', 'Conference Room', DATE '2021-11-05', 2);
INSERT INTO Equipment (EquipmentID, EquipmentName, Place, PurchaseDate, DeptID) VALUES
(3, 'Air Conditioner', 'Maintenance Room', DATE '2020-07-25', 3);
```

:Location

```
-- Insert into Location
INSERT INTO Location (LocationID, FloorID, AreaID, Availability) VALUES
(1, 1, 'North Wing', 'Available');
INSERT INTO Location (LocationID, FloorID, AreaID, Availability) VALUES
(2, 2, 'South Wing', 'Occupied');
INSERT INTO Location (LocationID, FloorID, AreaID, Availability) VALUES
(3, 3, 'East Wing', 'Available');
```

:MaintenanceRequest

```
-- Insert into MaintenanceRequest_
INSERT INTO MaintenanceRequest_ (RequestID, Urgency, DeptID, LocationID) VALUES
(1, 'High', 1, 1);
INSERT INTO MaintenanceRequest_ (RequestID, Urgency, DeptID, LocationID) VALUES
(2, 'Medium', 2, 2);
INSERT INTO MaintenanceRequest_ (RequestID, Urgency, DeptID, LocationID) VALUES
(3, 'Low', 3, 3);
```

:Task

```
-- Insert into Task
INSERT INTO Task (RequestID, StartDate, EndDate, Status, DeptID) VALUES
(1, DATE '2023-05-01', DATE '2023-05-05', 'Completed', 1);
INSERT INTO Task (RequestID, StartDate, EndDate, Status, DeptID) VALUES
(2, DATE '2023-06-10', DATE '2023-06-15', 'In Progress', 2);
INSERT INTO Task (RequestID, StartDate, EndDate, Status, DeptID) VALUES
(3, DATE '2023-07-20', DATE '2023-07-25', 'Pending', 3);
```

:select

```
-- Retrieve all rows from the Department table
SELECT * FROM Department;

-- Retrieve all rows from the Employee table
SELECT * FROM Employee;

-- Retrieve all rows from the Equipment table
SELECT * FROM Equipment;

-- Retrieve all rows from the Location table
SELECT * FROM Location;

-- Retrieve all rows from the MaintenanceRequest table
SELECT * FROM MaintenanceRequest;
```

שיטת data generator:

DEPARTMENT

<

Owner

Table

Number of records

>

C##HR

DEPARTMENT

400..1000

...

Name	Type	Size	Data	Master
DEPTID	NUMBER		Sequence(1, [Inc], [WithinParent]) + 0	
DEPTNAME	VARCHAR2	50	List('Electrical', 'Plumbing', 'HVAC', 'Carpentry', 'Painting', 'Landscaping', 'Appliance Repair', 'Ro ...	
BUDGET	NUMBER		Random(6000, 100000)	
MAXCAPACITY	NUMBER		Random(10, 100)	
CURRENTSIZE	NUMBER		Random(10, 100)	
*				

EMPLOYEE

< Owner

Table

Number of records

> C##HR

EMPLOYEE

1000





...

Name	Type	Size	Data	Master
EMPLOYEEID	NUMBER		Sequence(1, [Inc], [WithinParent])	...
NAME	VARCHAR2	50	FirstName + LastName	...
PHONENUMBER	VARCHAR2	20	'05' + Random(0, 9) + '-' + Random(100, 999) + '-' + Random(1000, 9999)	...
STARTWORK	DATE		Random('01-01-2000', '04-07-2024')	...
SALARY	NUMBER		Random(6000, 12000)	...
WORKINGHOURS	NUMBER		Random(5, 12)	...
▶ DEPTID	NUMBER		Random(1, 742)	...
*				...

שיטת mocoroco:

Equipment:

Equipment

Field Name	Type	Options
EquipmentID	Row Number	blank: 0 % Σ \times
EquipmentName	Custom List	hammer, screwdriver, pliers, wrench, tape measure, level, utility knife, saw, drill, chisel, Allen wrench (he: )
Place	Custom List	garage, workshop, tool shed, basement, attic, utility room, storage closet, tool chest, toolbox, pegboard, 
PurchaseDate	Datetime	01/01/2000  to 07/03/2024  format: m/d/yyyy \downarrow blank: 0 % Σ \times
DeptID	Number	min: 1 max: 742 decimals: 0 blank: 0 % Σ \times

Location_:

Location_

Field Name	Type	Options
LocationID	Row Number	blank: 0 % Σ \times
FloorID	Number	min: 1 max: 100
AreaID	Character Sequence	^###
Availability	Custom List	Available, occupied

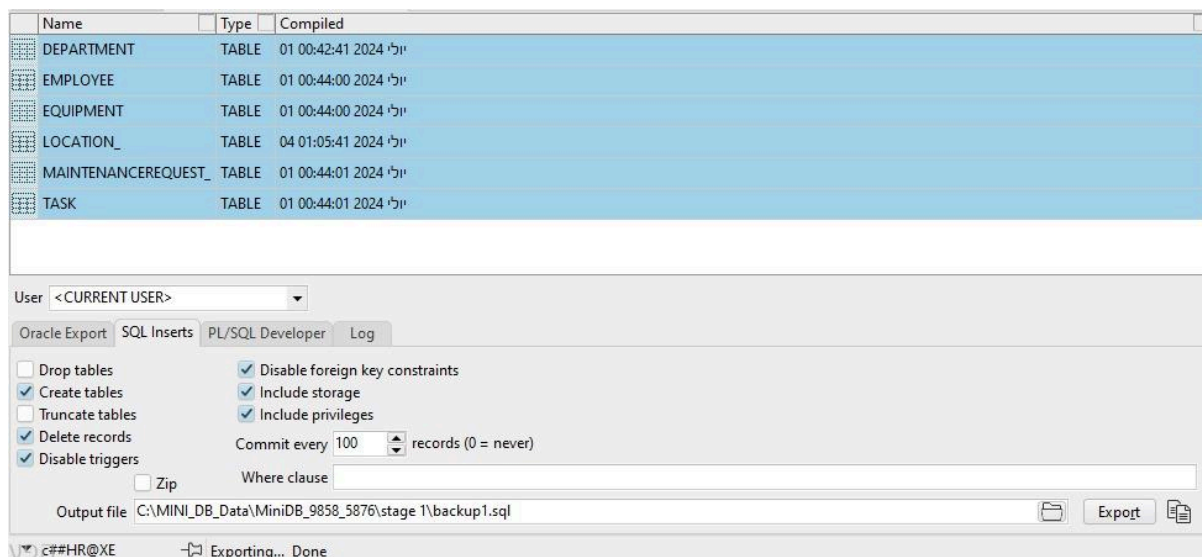
MaintenanceRequest:

Field Name	Type	Options
RequestID	Row Number	blank: 0 % Σ \times
Urgency	Custom List	Low, Medium, High
DeptID	Row Number	blank: 0 % Σ \times
LocationID	Row Number	blank: 0 % Σ \times

שיטת Excel files:

E	D	C	B	A	
DeptID	Status	EndDate	StartDate	RequestID	
1	Pending	05 01 2023	01 01 2023	1	2
2	Completed	06 01 2023	02 01 2023	2	3
3	InProgress	07 01 2023	03 01 2023	3	4
4		08 01 2023	04 01 2023	4	5

גיבוי:



```
-- Drop Location table
DROP TABLE Location_;

-- Drop MaintenanceRequest tabl
DROP TABLE MaintenanceRequest;

-- Drop Employee table
DROP TABLE Employee;

-- Drop Equipment table
DROP TABLE Equipment;

-- Drop Department table
DROP TABLE Department;

-- Drop Task table
DROP TABLE Task;
```

Oracle Import SQL Inserts PL/SQL Developer

☒ Analyze
☐ Commit
☒ Constraints
☒ Grants
☒ Ignore
☒ Indexes
☒ Rows
☐ Show

Statistics Always

Buffer size (KB) 30

From User

To User

Import Executable C:\app\admin\product\21c\dbf

Import file
C:\MINI_DB_Data\MiniDB_9858_5876\stage 1\backup1.sql

Import

דו"ח 2 - מיני פרויקט בבסיסי נתונים

רועי סאיג 327839858 - הראל ברנר

מערכת לניהול בתי מלון

לוגיסטיקה והזמנות ציוד

שאלות Select:

1. המנכ"ל רצה לדעת מה השכר הממוצע בכל מחלקה כדי לבדוק כמה כל מחלקה עולה לו ולקצץ עם צריך

```
SELECT Department.DeptID, Department.DeptName, AVG(Employee.Salary) AS AvgSalary
FROM Employee
JOIN Department ON Employee.DeptID = Department.DeptID
GROUP BY Department.DeptID, Department.DeptName
ORDER BY AvgSalary DESC
```

	DEPTID	DEPTNAME	AVGSALARY
1	3	Maintenance	50000
2	1	Engineering	25461.3333333333
3	2	Human Resources	24034
4	493	Roofing	11986
5	220	Appliance Repair	11902
6	721	HVAC	11896

2. המנכ"ל רוצה לבדוק האם יש משימות שתאריך הסיום המשוער שלהם עבר והמשימה עוד לא הסתיימה. משימות אלה צריכות להתבצע במהירות רבה מכיוון שלוקח כבר יותר מדי זמן לבצע אותן ממה ששיערו בהתחלה.

```
SELECT Task.RequestID, Task.StartDate, Task.EndDate, Task.Status, Department.DeptName
FROM Task
JOIN Department ON Task.DeptID = Department.DeptID
WHERE Task.EndDate < TO_DATE('01-07-2024', 'DD-MM-YYYY')
AND Task.Status != 'Completed';
```

	REQUESTID	STARTDATE	ENDDATE	STATUS	DEPTNAME
1	1	27 ????? 2023	04 ????? 2023	InProgress	Engineering
2	2	28 ????? 2023	25 ????? 2023	Pending	Human Resources
3	3	13 ????? 2024	02 ??? 2024	InProgress	Maintenance
4	4	16 ????? 2023	07 ????? 2023	Pending	Engineering
5	5	11 ??? 2024	10 ??? 2024	InProgress	Painting
6	6	12 ??? 2024	28 ????? 2023	Pending	Carpentry

3. המנהל רוצה לבדוק עבור כל מחלקה כמה עובדים יש בה ובכמה בקשות הם מטפלים כדי לבדוק האם כל העובדים נצרכים.

```
SELECT d.DeptName, COUNT(t.RequestID) AS TotalTasks, d.CurrentSize
FROM Department d
JOIN Task t ON d.DeptID = t.DeptID
GROUP BY d.DeptName, d.CurrentSize
ORDER BY TotalTasks DESC
```

	DEPTNAME	TOTALTASKS	CURRENTSIZE
1	Appliance Repair	4	37
2	Plumbing	3	65
3	Plumbing	3	78
4	Security Systems	3	69
5	Electrical	3	41
6	HVAC	3	53

4. המנכ"ל רוצה לדעת איזה מחלקות לא מאוישות כדי לדעת האם לבטל אותן ולחבר את המשימות שלהם למחלקות אחרות או לאייש אותן.

```
select *
FROM Department d
WHERE NOT EXISTS (
  SELECT 1
  FROM Employee e
  WHERE e.DeptID = d.DeptID
);
```

		DEPTID	DEPTNAME	BUDGET	MAXCAPACITY	CURRENTSIZE
1	15	Plumbing	...	26211	42	93
2	17	Roofing	...	81216	15	34
3	21	Fire Safety	...	43664	56	56
4	24	Plumbing	...	73467	100	20
5	34	Security Systems	...	83240	28	65
6	35	Painting	...	31301	59	47

שאלות Delete:

1. המנכ"ל רוצה למחוק את כל המשימות שהתבצעו ותאריך הסיום (המשוער) שלהם או לפני 12-12-2023 מכיוון שזה כבר ישן מדי ואין לשמור את זה.

```
DELETE FROM Task
WHERE Status = 'Completed'
AND EndDate < TO_DATE('12-12-2023', 'DD-MM-YYYY')
```

19:27 c##HR@XE [20:01:45] 66 rows deleted in 0.050 seconds

לפני:

```
select *
FROM Task
WHERE Status = 'Completed'
AND EndDate < TO_DATE('12-12-2023', 'DD-MM-YYYY')
```

	REQUESTID	STARTDATE	ENDDATE	STATUS	DEPTID
1	383	17/06/2024	09/12/2023	Completed	383
2	397	09/08/2023	21/08/2023	Completed	397
3	407	05/07/2024	23/08/2023	Completed	407
4	411	04/10/2023	30/09/2023	Completed	411
5	428	08/08/2023	27/07/2023	Completed	428
6	429	04/02/2024	21/07/2023	Completed	429
7	443	12/09/2023	28/10/2023	Completed	443
8	458	29/05/2024	28/07/2023	Completed	458
9	463	18/07/2023	03/12/2023	Completed	463
10	481	29/10/2023	10/12/2023	Completed	481
11	482	11/01/2024	25/11/2023	Completed	482
12	484	26/12/2023	04/09/2023	Completed	484
13	488	24/07/2023	23/11/2023	Completed	488
14	492	22/03/2024	25/09/2023	Completed	492
15	494	07/05/2024	31/07/2023	Completed	494
16	500	17/04/2024	29/10/2023	Completed	500
17	195	07/11/2023	07/08/2023	Completed	195
18	209	23/10/2023	02/12/2023	Completed	209

אחרי:

```
select *
FROM Task
WHERE Status = 'Completed'
AND EndDate < TO_DATE('12-12-2023', 'DD-MM-YYYY')
```

REQUESTID	STARTDATE	ENDDATE	STATUS	DEPTID
-----------	-----------	---------	--------	--------

2. המנכ"ל רוצה למצוא עובדים במחלקות בעלות תקציב נמוך, שהמשכורת שלהם גבוהה מהממוצע, ואין להם ציוד או בקשות תחזוקה או משימות תלויות, וכמות

העובדים במחלקה שלהם קרובה למיצוי הקיבולת של מספר העובדים. (זה יכול להיות מועיל למנהלים או למנהלי משאבי אנוש שמנסים לזהות בעיות או עובדים מסוימים שיכולים לדרוש תשומת לב מיוחדת בשל התנאים הייחודיים שלהם.)

```
DELETE FROM Employee
WHERE EmployeeID IN (
  SELECT e.EmployeeID
  FROM Employee e
  JOIN Department d ON e.DeptID = d.DeptID
  LEFT JOIN Equipment eq ON eq.DeptID = d.DeptID
  LEFT JOIN MaintenanceRequest mr ON mr.DeptID = d.DeptID
  LEFT JOIN Task t ON t.DeptID = d.DeptID
  WHERE d.Budget < 100000
  AND e.Salary > (SELECT AVG(Salary) FROM Employee)
  AND eq.EquipmentID IS NULL
  AND mr.RequestID IS NULL
  AND t.RequestID IS NULL
  AND d.CurrentSize > d.MaxCapacity * 0.8
);
```

34:1 c##HR@XE [14:19:05] 3 rows deleted in 0.097 seconds

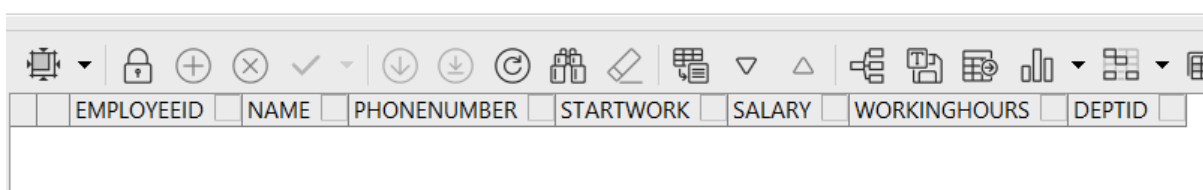
לפני:

```
select *
FROM Employee
WHERE EmployeeID IN (
  SELECT e.EmployeeID
  FROM Employee e
  JOIN Department d ON e.DeptID = d.DeptID
  LEFT JOIN Equipment eq ON eq.DeptID = d.DeptID
  LEFT JOIN MaintenanceRequest mr ON mr.DeptID = d.DeptID
  LEFT JOIN Task t ON t.DeptID = d.DeptID
  WHERE d.Budget < 100000
  AND e.Salary > (SELECT AVG(Salary) FROM Employee)
  AND eq.EquipmentID IS NULL
  AND mr.RequestID IS NULL
  AND t.RequestID IS NULL
  AND d.CurrentSize > d.MaxCapacity * 0.8
);
```

	EMPLOYEEID	NAME	PHONENUMBER	STARTWORK	SALARY	WORKINGHOURS	DEPTID
1	10	TobeyWright	055-488-5728	02/07/2011	11668	7	724
2	789	JudgePatton	051-894-7708	27/08/2021	11143	6	711
3	50	ElleLiu	051-962-4749	08/04/2006	10286	6	739

אחרי:

```
select *
FROM Employee
WHERE EmployeeID IN (
    SELECT e.EmployeeID
    FROM Employee e
    JOIN Department d ON e.DeptID = d.DeptID
    LEFT JOIN Equipment eq ON eq.DeptID = d.DeptID
    LEFT JOIN MaintenanceRequest_ mr ON mr.DeptID = d.DeptID
    LEFT JOIN Task t ON t.DeptID = d.DeptID
    WHERE d.Budget < 100000
    AND e.Salary > (SELECT AVG(Salary) FROM Employee)
    AND eq.EquipmentID IS NULL
    AND mr.RequestID IS NULL
    AND t.RequestID IS NULL
    AND d.CurrentSize > d.MaxCapacity * 0.8
);
```



EMPLOYEEID	NAME	PHONENUMBER	STARTWORK	SALARY	WORKINGHOURS	DEPTID
------------	------	-------------	-----------	--------	--------------	--------

שאלות: Update

1. בחודשים האחרונים העובדים בכל המחלקות עבדו ממש טוב והמנכל רוצה לצפר את העובדים ולתת העלאות שכר במחלקות השונות, מבלי לחרוג מתקציב המחלקה. הוא רוצה לראות באיזה מחלקות אפשר להעלות את המשכורות של העובדים בלי לחרוג מתקציב המחלקה.

```
UPDATE Employee
SET Salary = Salary * 1.10 -- Increase salary by 10%
WHERE DeptID = (
    SELECT DeptID
    FROM Department
    WHERE DeptID = Employee.DeptID
    AND Budget >= (
        SELECT SUM(Salary) * 1.10
        FROM Employee
        WHERE DeptID = Department.DeptID
    )
);
```

22:34 c##HR@XE [18:02:39] 793 rows updated in 0.178 seconds

לפני:

23

```
select *
from Employee
--SET Salary = Salary * 1.10 -- Increase salary by 10%
WHERE DeptID = (
    SELECT DeptID
    FROM Department
    WHERE DeptID = Employee.DeptID
    AND Budget >= (
        SELECT SUM(Salary) * 1.10
        FROM Employee
        WHERE DeptID = Department.DeptID
    )
);
```

	EMPLOYEEID	NAME	PHONENUMBER	STARTWORK	SALARY	WORKINGHOURS	DEPTID
1	946	DanDayne	050-742-4444	11/03/2013	6252	10	129
2	947	DennisSingletary	053-139-3783	24/02/2013	11571	6	736
3	948	EttaArden	052-408-4192	17/06/2008	7942	8	677
4	949	KevinRippy	056-291-4552	07/09/2013	9122	10	92
5	950	ElisabethSerbedzija	056-614-6013	15/01/2003	11196	12	562
6	951	DennisDupree	058-267-7043	14/10/2001	9787	11	328

אחר:

```
select *
from Employee
--SET Salary = Salary * 1.10 -- Increase salary by 10%
WHERE DeptID = (
    SELECT DeptID
    FROM Department
    WHERE DeptID = Employee.DeptID
    AND Budget >= (
        SELECT SUM(Salary) * 1.10
        FROM Employee
        WHERE DeptID = Department.DeptID
    )
);
```

	EMPLOYEEID	NAME	PHONENUMBER	STARTWORK	SALARY	WORKINGHOURS	DEPTID
1	946	DanDayne	050-742-4444	11/03/2013	6877	10	129
2	947	DennisSingletary	053-139-3783	24/02/2013	12728	6	736
3	948	EttaArden	052-408-4192	17/06/2008	8736	8	677
4	949	KevinRippy	056-291-4552	07/09/2013	10034	10	92
5	950	ElisabethSerbedzija	056-614-6013	15/01/2003	12316	12	562
6	951	DennisDupree	058-267-7043	14/10/2001	10766	11	328

2. המנכ"ל רוצה לעדכן את הדחיפות של כל המשימות שזמן הסיום המשוער שלהם עבר בדצמבר 2023 להיות HIGH, כלומר שצריך לעבוד עליהם כמה שיותר מהר.

```
UPDATE MaintenanceRequest
SET Urgency = 'High'
WHERE RequestID IN (
    SELECT RequestID
    FROM Task
    WHERE EndDate < TO_DATE('01-12-2023', 'DD-MM-YYYY')
);
```

63:6 c##HR@XE [14:11:02] 193 rows updated in 0.057 seconds

לפני:

```
select * from MaintenanceRequest_
--SET Urgency = 'High'
WHERE RequestID IN (
    SELECT RequestID
    FROM Task
    WHERE EndDate < TO_DATE('01-12-2023', 'DD-MM-YYYY')
);
```

	REQUESTID	URGENCY	DEPTID	LOCATIONID
1	382	Low	237	937
2	394	Low	337	788
3	395	High	75	883
4	402	Medium	736	120
5	405	Medium	117	731
6	408	Medium	612	242

אחרי:


```
select * from MaintenanceRequest_
--SET Urgency = 'High'
WHERE RequestID IN (
    SELECT RequestID
    FROM Task
    WHERE EndDate < TO_DATE('01-12-2023', 'DD-MM-YYYY')
);
```

	REQUESTID	URGENCY	DEPTID	LOCATIONID
1	382	High	237	937
2	394	High	337	788
3	395	High	75	883
4	402	High	736	120
5	405	High	117	731
6	408	High	612	242

שאלות עם פרמטרים:

1. המנכ"ל רצה רשימה של כל הצידוד שנקנה בטווח תאריכים מסוים (שיוכנס כפרמטר).

```
select * from Equipment t
where PurchaseDate
between date &<name=Purchase1 type=string> and date &<name=Purchase2 type=string>
```

Variables

Name	Value
Purchase1	2018-01-01
Purchase2	2024-01-01

OK

Cancel

Clear

	EQUIPMENTID	EQUIPMENTNAME	PLACE	PURCHASEDATE	DEPTID
1	1	Laptop	Office	15 ?????? 2022	1
2	2	Projector	Conference Room	05 ?????? 2021	2
3	3	Air Conditioner	Maintenance Room	25 ???? 2020	3
4	7	wire strippers	utility trailer	28 ?????? 2022	7
5	11	crowbar	workbench	04 ?????? 2021	11
6	13	pliers	corner cabinet	08 ?????? 2023	13
7	18	jigsaw	utility trailer	01 ?????? 2021	18
8	21	bench grinder	equipment shed	08 ?????? 2021	21

2. המנכ"ל רצה שתהיה לו אפשרות לראות את כל העובדים שעובדים מספר שעות מסוים ואת משכורתם כדי לראות שכולם מקבלים שכר הגיוני ולצפר עובדים במידת הצורך.

```
SELECT e.EmployeeID, e.Name, e.salary, e.workinghours
FROM Employee e
WHERE WorkingHours = &<name = "Hours" hint="hours value between 5-15">
```

	EMPLOYEEID	NAME	SALARY	WORKINGHOURS
1	15	Jean-LucFeore	9457	6
2	17	FredericSylvian	10979	6
3	44	GrantApple	10119	6
4	47	GeggyDzundza	11073	6
5	59	DebiKattan	8228	6
6	65	ChazzDaniels	9089	6
7	66	KazemVoight	10788	6

3. המנכ"ל רצה שתהיה לו אפשרות לראות את כל בקשות התחזוקה לפי רמת דחיפות מסוימת

```
SELECT RequestID, LocationID
FROM MaintenanceRequest
WHERE Urgency = &<name = "level" list="High, Medium, Low" type="string">;
```

Variables	
Name	Value
level	Medium

	REQUESTID	LOCATIONID
1	8	453
2	9	691
3	10	782
4	11	500
5	15	972
6	18	327
7	25	194

4. כשעובד צריך כלי מסוים בשביל לטפל בתקלה. הוא צריך שתהיה לו אפשרות לחפש כלי לפי שם הכלי ומספר המחלקה שלו ולבדוק האם הכלי קיים

```

/*UPDATE MaintenanceRequest
SET Urgency = 'High'
WHERE RequestID IN (
    SELECT RequestID
    FROM Task
    WHERE EndDate < TO_DATE('01-12-2023',
);*/

SELECT e.Place
FROM Equipment e
JOIN Department d ON e.DeptID = d.DeptID
WHERE d.DeptName = '&Department'
AND e.EquipmentName = '&Equipment';

```

Variables

Name	Value
Department	Engineering
Equipment	Laptop

OK

Cancel

Clear

	PLACE
1	Office

אילוצים:

אילוץ ראשון:

לפני הכנסת משכורת לעובד, עלינו לבדוק שהמשכורת גדולה מ0. כי אין דבר כזה משכורת שלילית.

```

ALTER TABLE Employee
ADD CONSTRAINT Check_Salary CHECK (Salary > 0);

```

1:13

c##HR@XE

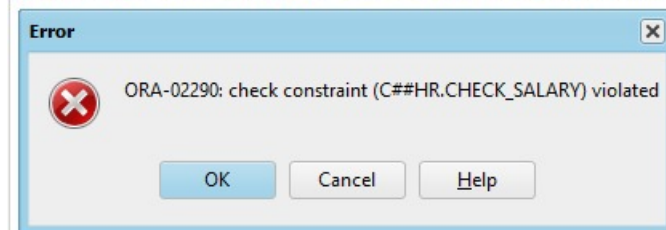
[14:47:36] Done in 0.074 seconds

זה באמת נוסף לנו:

Name	Condition	Enabled	Deferrable	Deferred	Validated	Last change
▶ CHECK_SALARY	Salary > 0	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	יולי 12 14:47:36 2024
PHONE_FORMAT	REGEXP_LIKE(PhoneNumber, '^(\d{3}-\d{3}-\d{4})\$')	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	יולי 01 00:44:00 2024

ננסה להכניס עובד עם משכורת 0. ונראה שזה זורק לנו ERROR.

```
INSERT INTO Employee (EmployeeID, Name, PhoneNumber, StartWork, Salary, WorkingHours, DeptID)
VALUES (2, 'Jane Smith', '987-654-3210', TO_DATE('01-02-2022','DD-MM-YYYY'), 0, 40, 2);
```



אילוף שני:

ערך ברירת המחדל של האם המקום זמין או לא הוא שהוא זמין, כלומר availabel.

```
ALTER TABLE Location_
MODIFY Availability VARCHAR(10) DEFAULT 'Available';
```

000 & 2:50 c##HR@XE [14:53:55] Done in 0.105 seconds

באמת נוסף לנו:

Name	Virtual	Type	Nullable	Default/Expr.	Generated	On Null	Invisible	Storage	Comments
▶ LOCATIONID	<input type="checkbox"/>	INTEGER	<input type="checkbox"/>			<input type="checkbox"/>	<input type="checkbox"/>		
FLOORID	<input type="checkbox"/>	INTEGER	<input type="checkbox"/>			<input type="checkbox"/>	<input type="checkbox"/>		
AREAIID	<input type="checkbox"/>	VARCHAR2(15)	<input type="checkbox"/>			<input type="checkbox"/>	<input type="checkbox"/>		
AVAILABILITY	<input type="checkbox"/>	VARCHAR2(10)	<input type="checkbox"/>	'Available'		<input type="checkbox"/>	<input type="checkbox"/>		

נכניס מיקום ללא ערך בזמינות:

```
INSERT INTO Location_ (LocationID, FloorID, AreaID)
VALUES (1001, 3, 'B456');
```

000 & 2:26 c##HR@XE [16:05:02] 1 row inserted in 0.020 seconds

אנחנו רואים שהוא באמת נכנס בתור "available".

1000	1000	24	K282	Available
▶ 1001	1001	3	B456	Available

אילוצ שלישי:

אם תאריך הרכישה של פריט מסוים הוא לאחר התאריך הנוכחי, זה לא הגיוני. ולכן נבדוק שתאריך הרכישה בהכרח לפני התאריך הנוכחי.

```
ALTER TABLE Equipment
ADD CONSTRAINT Check_PurchaseDate CHECK (PurchaseDate <= TO_DATE('07-07-2024', 'DD-MM-YYYY'));
```

2:13 c##HR@XE [15:15:43] Done in 0.336 seconds

באמת נוסף לנו:

Name	Condition	Enabled	Deferrable	Deferred	Validated	Last change
▶ CHECK_PURCHASEDATE	PurchaseDate <= TO_DATE('07-07-2024', 'DD-MM-YYYY')	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	12 יולי 2024 15:15:43

נכניס תאריך עתידי בזמן הרכישה ונראה כי תזרק לנו שגיאה:

```
INSERT INTO Equipment (EquipmentID, EquipmentName, Place, PurchaseDate, DeptID)
VALUES (2, 'Projector', 'Conference Room B', TO_DATE('08-08-2024', 'DD-MM-YYYY'), 2);
```

Error

ORA-02290: check constraint (C##HR.CHECK_PURCHASEDATE) violated

OK Cancel Help

דו"ח 3 - מיני פרויקט בבסיסי נתוניםרועי סאיג 327839858 - הראל ברנרמערכת לניהול בתי מלוןלוגיסטיקה והזמנות ציוד

פונקציה 1 - הפונקציה בודקת מי המחלקה שממוצע שעות העבודה של העובדים שלה הוא הכי גדול. את המחלקה הזאת המנכ"ל רוצה לצפר ולכן הוא מעלה את המשכורת של כולם בפי 1.25, ומעדכן את תקציב המחלקה שיאים למשכורות המעודכנות.

הנתונים לפני ההרצה:

המחלקה שממוצע שעות העבודה של העובדים שלה הוא הכי גדול והתקציב שלה.

```

SELECT *
FROM Department
WHERE DeptID = (
    SELECT DeptID
    FROM (
        SELECT DeptID, AVG(WorkingHours) AS AvgWorkingHours
        FROM Employee
        GROUP BY DeptID
    ) AvgDeptWorkingHours
    WHERE AvgWorkingHours = (
        SELECT MAX(AvgWorkingHours)
        FROM (
            SELECT AVG(WorkingHours) AS AvgWorkingHours
            FROM Employee
            GROUP BY DeptID
        ) MaxAvgWorkingHours
    )
);

```

	DEPTID	DEPTNAME	BUDGET	MAXCAPACITY	CURRENTSIZE
▶	1	3 Maintenance	75000	20	15

משכורות העובדים של המחלקה שממוצע שעות העבודה של העובדים שלה הוא הכי גדול.

```
SELECT *
FROM Employee
WHERE DeptID = (
    SELECT DeptID
    FROM (
        SELECT DeptID, AVG(WorkingHours) AS AvgWorkingHours
        FROM Employee
        GROUP BY DeptID
    ) AvgDeptWorkingHours
    WHERE AvgWorkingHours = (
        SELECT MAX(AvgWorkingHours)
        FROM (
            SELECT AVG(WorkingHours) AS AvgWorkingHours
            FROM Employee
            GROUP BY DeptID
        ) MaxAvgWorkingHours
    )
);
```

	EMPLOYEEID	NAME	PHONENUMBER	STARTWORK	SALARY	WORKINGHOURS	DEPTID
1	3	Emily Johnson	345-678-9012	20/03/2019	55000	40	3
2	1601	Russell Sleight	959-737-5100	04/08/2007	32022	11	3
3	3001	John Doe	555-123-4567	15/01/2023	60000	40	3
4	3002	Jane Smith	555-987-6543	20/02/2023	50000	35	3

קוד הפונקציה:

```
CREATE OR REPLACE FUNCTION UpdateBudgetForTopWorkingHoursDept
RETURN INT AS
    newBudget INT;
    topDeptID INT;
    r INT;
    --totalSalary INT;
    --totalEquipmentCost INT;
BEGIN

    SELECT DeptID
    INTO topDeptID
    FROM (
        SELECT DeptID, AVG(WorkingHours) AS AvgWorkingHours
        FROM Employee
        GROUP BY DeptID
        ORDER BY AvgWorkingHours DESC
        FETCH FIRST 1 ROWS ONLY
    );

    r := '&raise';
    UPDATE Employee
    SET Salary = Salary * r
    WHERE DeptID = topDeptID;

    SELECT SUM(Salary)
    INTO newBudget
    FROM Employee
    WHERE DeptID = topDeptID;
```

```

UPDATE Department
SET Budget = newBudget
WHERE DeptID = topDeptID;

RETURN newBudget;
END;

```

תוצאה (תקציב מעודכן):

```
Updated Budget: 109078
```

עדכון תקציב המחלקה (התקציב גדל):

	DEPTID	DEPTNAME	BUDGET	MAXCAPACITY	CURRENTSIZE
▶ 1	3	Maintenance	109078	20	15

עדכון משכורות העובדים (המשכורות של כל העובדים גדלו):

	EMPLOYEEID	NAME	PHONENUMBER	STARTWORK	SALARY	WORKINGHOURS	DEPTID
▶ 1	3	Emily Johnson	345-678-9012	20/03/2019	85938	40	3
2	1601	Russell Sleight	959-737-5100	04/08/2007	50035	11	3
3	3001	John Doe	555-123-4567	15/01/2023	75000	40	3
4	3002	Jane Smith	555-987-6543	20/02/2023	62500	35	3

פורצדורה 1 - תחילה אנחנו סופרים את מספר העובדים בכל מחלקה ומעדכנים את נתון ה-CurrentSize. לאחר מכן אנחנו עוברים בלולאה על כל המחלקות שמספר העובדים שלהם גדול מהמכסה של העובדים שהמחלקה יכולה להחזיק (האם $CurrentSize > MaxCapacity$). עבור כל מחלקה כזאת, אנחנו נכנסים לעוד לולאה שעד שמספר העובדים במחלקה קטן מהמכסה, הוא מוחק עובדים מהמחלקה (עובד אחד בכל איטרציה (את העובד הכי ותיק של המחלקה)) ומעדכן את שדה ה-CurrentSize של המחלקה.

המחלקות ושדה ה-CurrentSize לפני העדכון:


```
select *
from department
```

	DEPTID	DEPTNAME	BUDGET	MAXCAPACITY	CURRENTSIZE
1	709	Painting	40854	75	87
2	710	Landscaping	58464	93	24
3	711	Carpentry	74622	88	91
4	712	HVAC	80978	72	72
5	713	Appliance Repair	66077	97	88
6	714	Fire Safety	14431	30	29
7	715	Appliance Repair	40229	62	31
8	716	Roofing	20297	28	32
9	717	Landscaping	14654	64	73
10	718	Security Systems	19841	37	78

העובדים לפני המחיקות (כרגע יש 2999 עובדים):

```
select *
from employee
```

	EMPLOYEEID	NAME	PHONENUMBER	STARTWORK	SALARY	WORKINGHOURS	DEPTID
1	1458	Aile Chawkey	770-681-9665	23/08/2020	40515	12	625
2	1459	Bird Dumbrill	195-847-6837	09/12/2005	8470	6	437
3	1460	Leigh Scudders	922-218-4160	17/11/2013	22033	8	284
4	1461	Shoshana Scarratt	243-651-8597	05/08/2023	79060	8	380
5	1462	Marj De Gregario	352-649-5596	18/03/2024	64869	11	562
6	1463	Earl Ratchford	106-560-7857	13/02/2015	11765	7	246
7	1464	Arabele Dies	248-823-4408	19/05/2017	30985	5	532
8	1465	Hazlett Eshelby	781-176-7756	22/04/2016	5591	7	440
9	1466	Adriane Mackerness	821-592-3158	29/01/2020	40134	5	346
10	1467	Mommy Widdall	728-344-3376	02/05/2017	35311	11	358

1 of 2999 0:02 c##HR@XE [16:15:29] 2999 rows selected in 2.714 seconds

קוד הפרוצדורה:

```

CREATE OR REPLACE PROCEDURE UpdateAndAdjustDepartmentSizes IS
    depID INT;
    current_size INT;
    max_capacity INT;
BEGIN

    UPDATE Department d
    SET CurrentSize = (
        SELECT COUNT(*)
        FROM Employee e
        WHERE e.DeptID = d.DeptID
    );

    FOR cur IN (
        SELECT DeptID
        FROM Department
        WHERE CurrentSize > MaxCapacity
    ) LOOP
        depID := cur.DeptID;

        -----
        LOOP

            SELECT CurrentSize, MaxCapacity
            INTO current_size, max_capacity
            FROM Department
            WHERE DeptID = depID;

            EXIT WHEN current_size <= max_capacity;

            DELETE FROM Employee
            WHERE EmployeeID = (
                SELECT EmployeeID
                FROM Employee
                WHERE DeptID = depID
                ORDER BY StartWork ASC
                FETCH FIRST 1 ROWS ONLY
            );

            UPDATE Department
            SET CurrentSize = CurrentSize - 1
            WHERE DeptID = depID;
        END LOOP;
    END LOOP;
END UpdateAndAdjustDepartmentSizes;

```

עדכון ה-CurrentSize לאחר הרצת הפרוצדורה:

	DEPTID	DEPTNAME	BUDGET	MAXCAPACITY	CURRENTSIZE
1	709	Painting	40854	75	1
2	710	Landscaping	58464	4	4

רשימת העובדים לאחר הרצת הפרוצדורה (עכשיו יש רק 2997 עובדים):

```
select *
from employee
```

	EMPLOYEEID	NAME	PHONENUMBER	STARTWORK	SALARY	WORKINGH
1	1458	Aile Chawkley	770-681-9665	23/08/2020	40515	
2	1459	Bird Dumbrill	195-847-6837	09/12/2005	8470	
3	1460	Leigh Scudders	922-218-4160	17/11/2013	22033	
4	1461	Shoshana Scarratt	243-651-8597	05/08/2023	79060	
5	1462	Marj De Gregario	352-649-5596	18/03/2024	64869	
6	1463	Earl Ratchford	106-560-7857	13/02/2015	11765	
7	1464	Arabele Dies	248-823-4408	19/05/2017	30985	
8	1465	Hazlett Eshelby	781-176-7756	22/04/2016	5591	
9	1466	Adriane Mackerness	821-592-3158	29/01/2020	40134	
10	1467	Mommy Widdall	728-344-3376	02/05/2017	35311	

252:14 0:02 c##HR@XE [16:29:04] 2997 rows selected in 2.033 seconds

פונקציית main - הפונקציה מריצה את שתי הפונקציות:

```
DECLARE
    updatedBudget INT;
BEGIN

    UpdateAndAdjustDepartmentSizes;
    |
    updatedBudget := UpdateBudgetForTopWorkingHoursDept();

    DBMS_OUTPUT.PUT_LINE('Updated Budget: ' || updatedBudget);

END;
```

procedure2:

הפרוצדורה שכתבנו עושה את הדבר הבא:
 היא מוצאת את כל המשימות, שצריכות להתבצע, כלומר את כל
 MaintenanceRequest, ושולחת אותם לביצוע, כלומר מתאימה להם Task.
 כל משימה שלו נשלחה לביצוע, כעת נשלחת לביצוע, עם תאריך התחלה של התאריך
 הנוכחי, ועם תאריך סיום לעוד שבועיים, כלומר יש לעובדים שבועיים לבצע את
 המשימה הזו. כמובן שהמשימה תוגדר בהתחלה בתור "Pending", כלומר ממתינה
 לכך שמישהו יבצע אותה.
 ולכן אנחנו אמורים לבדוק לפני הפעלת הפרוצדורה, מה הם המשימות שלא נשלחו
 לביצוע, בשביל להראות שהפרוצדורה באמת פעלה, ושהיא שלחה את כל המשימות
 לביצוע.

```
SELECT *
FROM MaintenanceRequest_ m
WHERE NOT EXISTS (
    SELECT 1
    FROM Task t
    WHERE t.RequestID = m.RequestID
);
```

זו הבדיקה:
 ולמטה אנו רואים שישנן
 הרבה משימות שלא
 נשלחו לביצוע עוד.

		REQUESTID	URGENCY	DEPTID	LOCATIONID
▶	1	989	Low	281	814
	2	990	High	164	944
	3	991	High	318	638
	4	992	Medium	249	634
	5	993	Low	451	690
	6	994	High	405	849
	7	995	High	707	855
	8	996	High	351	280
	9	997	Medium	8	161
	10	998	High	732	350

זו הפרוצדורה:

```
create or replace procedure AssignTasksToMaintenanceRequests is
begin
  FOR req IN (
    SELECT RequestID, DeptID
    FROM MaintenanceRequest_
    WHERE RequestID NOT IN (SELECT RequestID FROM Task)
  ) LOOP
    INSERT INTO Task (RequestID, StartDate, EndDate, Status, DeptID)
    VALUES (req.RequestID, TO_DATE('2024-07-14', 'YYYY-MM-DD'),
      TO_DATE('2024-07-28', 'YYYY-MM-DD'), 'Pending', req.DeptID);
  END LOOP;

  COMMIT;
EXCEPTION
  WHEN OTHERS THEN
    ROLLBACK;
    DBMS_OUTPUT.PUT_LINE('Error occurred: ' || SQLERRM);
end AssignTasksToMaintenanceRequests;
```

נזמין אותה:

```
BEGIN
  -- Call AssignTasksToMaintenanceRequests procedure
  BEGIN
    AssignTasksToMaintenanceRequests;
  END;
```

000 & 30:1 c##HR@XE [16:38:28] Done in 0.204 seconds

נבדוק שוב את הבדיקה שעשינו למעלה, ונראה שלא קיימת משימה שלא נשלחה לביצוע, כלומר שלכל MaintenanceRequest חובר Task, וזה אומר שהפרוצדורה הצליחה:

```
SELECT *
FROM MaintenanceRequest_ m
WHERE NOT EXISTS (
  SELECT 1
  FROM Task t
  WHERE t.RequestID = m.RequestID
);
```

REQUESTID URGENCY DEPTID LOCATIONID

function2:

הפונקציה, סוג של ממשיכה את הפרוצדורה:

בחלקה הראשון של הפונקציה הפונקציה ממיינת את כל המשימות שנשלחו לביצוע לפי דחיפות ואז לפי הדדליין המוקדם יותר. לאחר מכן היא סופרת את מספר העובדים בכל מחלקה ושומרת את זה בתוך מערך הסוציאטיבי.

ואז הפונקציה עוברת על כל מחלקה ועבור כל עובד אם יש משימה פנויה, היא מורידה את העובד ואת המשימה. וזה בעצם אומר שהעובד הזה לא פנוי כי הוא עסוק במשימה שקיבל.

```

1 CREATE OR REPLACE FUNCTION AssignTasksAndUpdateEmployees
2 RETURN INT
3 IS
4     CURSOR task_cursor IS
5         SELECT t.RequestID, t.DeptID
6         FROM Task t
7         JOIN MaintenanceRequest_m ON t.RequestID = m.RequestID
8         ORDER BY m.Urgency DESC, t.EndDate ASC;
9
10    TYPE DeptEmployeeCount IS TABLE OF INT INDEX BY PLS_INTEGER; -- integers associative array
11    available_employees DeptEmployeeCount;
12    remaining_tasks INT := 0;
13 BEGIN
14
15    FOR dept_rec IN (SELECT DeptID, COUNT(*) AS EmployeeCount FROM Employee GROUP BY DeptID) LOOP -- count the number of employees on each department
16        available_employees(dept_rec.DeptID) := dept_rec.EmployeeCount; -- each place in the array is a number of employees in one department
17    END LOOP;
18
19
20    FOR task_rec IN task_cursor LOOP
21        IF available_employees.EXISTS(task_rec.DeptID) AND available_employees(task_rec.DeptID) > 0 THEN
22            available_employees(task_rec.DeptID) := available_employees(task_rec.DeptID) - 1;
23        ELSE
24            remaining_tasks := remaining_tasks + 1;
25        END IF;
26    END LOOP;

```

בחלקה השני הפונקציה דואגת לכך שאם יש עובדים שלא קיבלו שום משימה שעות העבודה שלהם יתעדכנו כ0. כי הם בעצם לא עובדים כרגע.

```

28
29    FOR dept_id IN available_employees.FIRST..available_employees.LAST LOOP -- iterate on the associative array
30        IF available_employees.EXISTS(dept_id) AND available_employees(dept_id) > 0 THEN -- if there are employees that dont work now
31            UPDATE Employee
32            SET WorkingHours = 0
33            WHERE DeptID = dept_id
34            AND EmployeeID IN (
35                SELECT EmployeeID
36                FROM (
37                    SELECT EmployeeID
38                    FROM Employee
39                    WHERE DeptID = dept_id
40                    AND ROWNUM <= available_employees(dept_id)
41                )
42            );
43        END IF;
44    END LOOP;
45
46    RETURN remaining_tasks;
47 EXCEPTION
48 WHEN OTHERS THEN
49     DBMS_OUTPUT.PUT_LINE('Error occurred: ' || SQLERRM);
50     RETURN NULL;
51 END;

```

לפני שנריץ את הפונקציה, נבדוק כמה עובדים עם מספר שעות עבודה 0 יש במערכת:
נראה אין כאלו. לאחר שנריץ את הפונקציה, צריכים להיות כמה כאלו אם יש יותר עובדים ממשימות.

```
select *
from employee
where workinghours = 0
```

REQUESTID	URGENCY	DEPTID	LOCATIONID
-----------	---------	--------	------------

נריץ את הפונקציה ונראה אם זה עבד:

```
select *
from employee
where workinghours = 0
```

EMPLOYEEID	NAME	PHONENUMBER	STARTWORK	SALARY	WORKINGHOURS	DEPTID
1	Aile Chawkey	770-681-9665	23/08/2020	40515	0	625
2	Leigh Scudders	922-218-4160	17/11/2013	22033	0	284
3	Shoshana Scarratt	243-651-8597	05/08/2023	79060	0	380
4	Marj De Gregario	352-649-5596	18/03/2024	64869	0	562
5	Earl Ratchford	106-560-7857	13/02/2015	11765	0	246
6	Arabele Dies	248-823-4408	19/05/2017	30985	0	532
7	Hazlett Eshelby	781-176-7756	22/04/2016	5591	0	440
8	Adriane Mackerness	821-592-3158	29/01/2020	40134	0	346
9	Mommy Widdall	728-344-3376	02/05/2017	35311	0	358
10	Barby Rasmus	606-963-7230	03/08/2000	80901	0	406

אנחנו רואים שכעת ישנם הרבה עובדים ששעות העבודה שלהם זה 0. כלומר יש יותר עובדים ממשימות והפונקציה באמת עובדת.

main2:

```
BEGIN
  BEGIN
    AssignTasksToMaintenanceRequests;
  END;

  BEGIN
    DECLARE
      remaining_tasks INT;
    BEGIN
      remaining_tasks := AssignTasksAndUpdateEmployees;
      DBMS_OUTPUT.PUT_LINE('Number of remaining tasks: ' || remaining_tasks);
    END;
  END;
END;
```

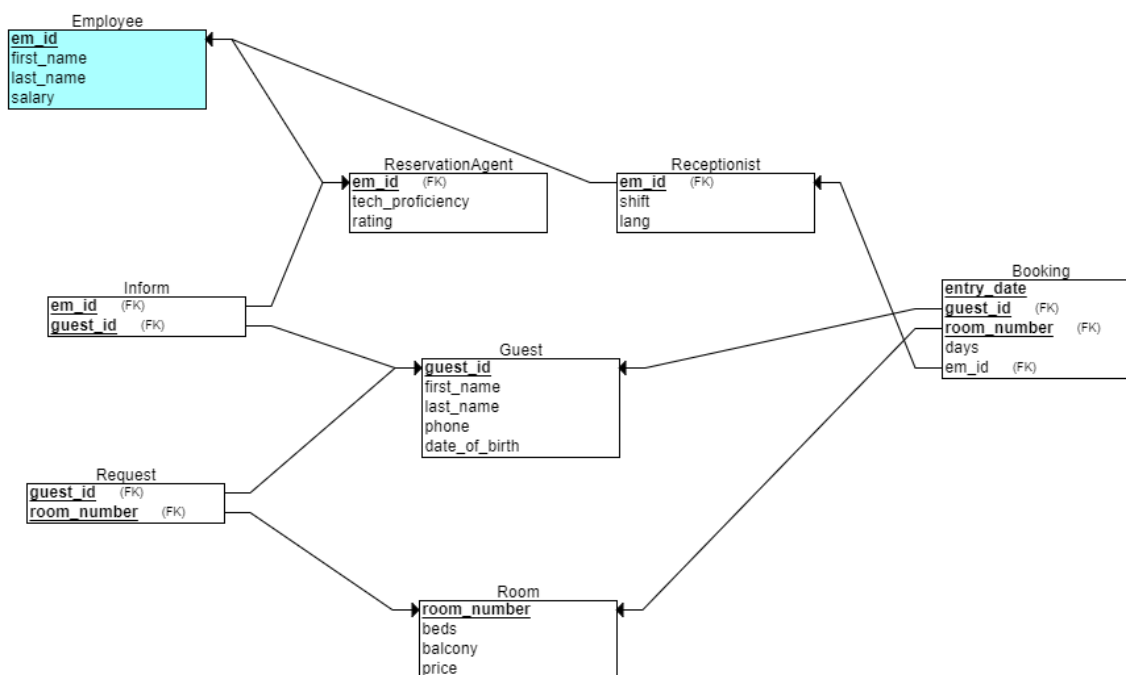

דו"ח 4 - מיני פרויקט בבסיסי נתונים

רועי סאיג 327839858 - הראל ברנר

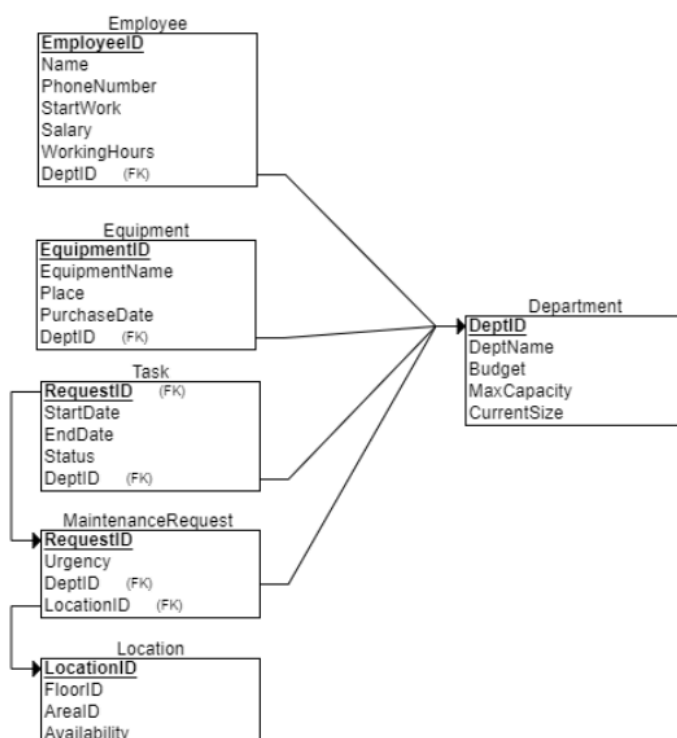
מערכת לניהול בתי מלון

לוגיסטיקה והזמנות ציוד

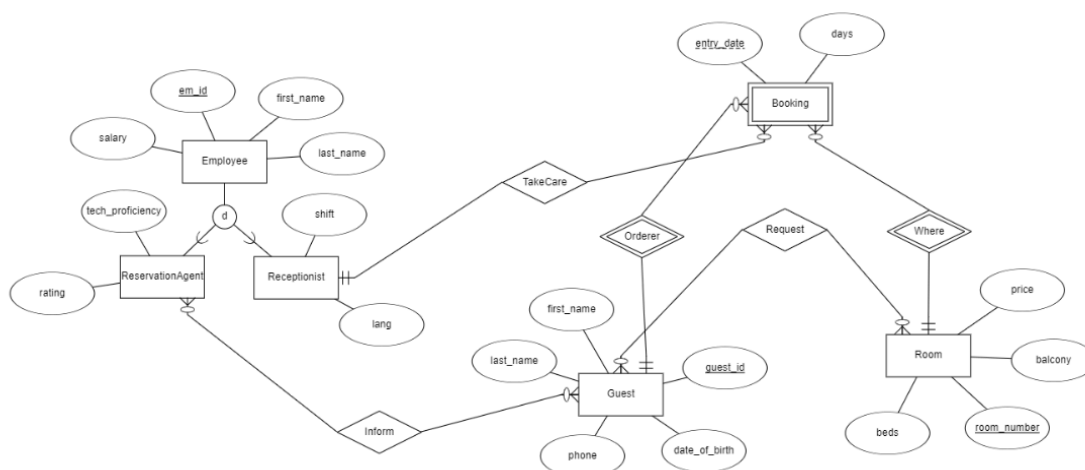
תרשים DSD שלהם:



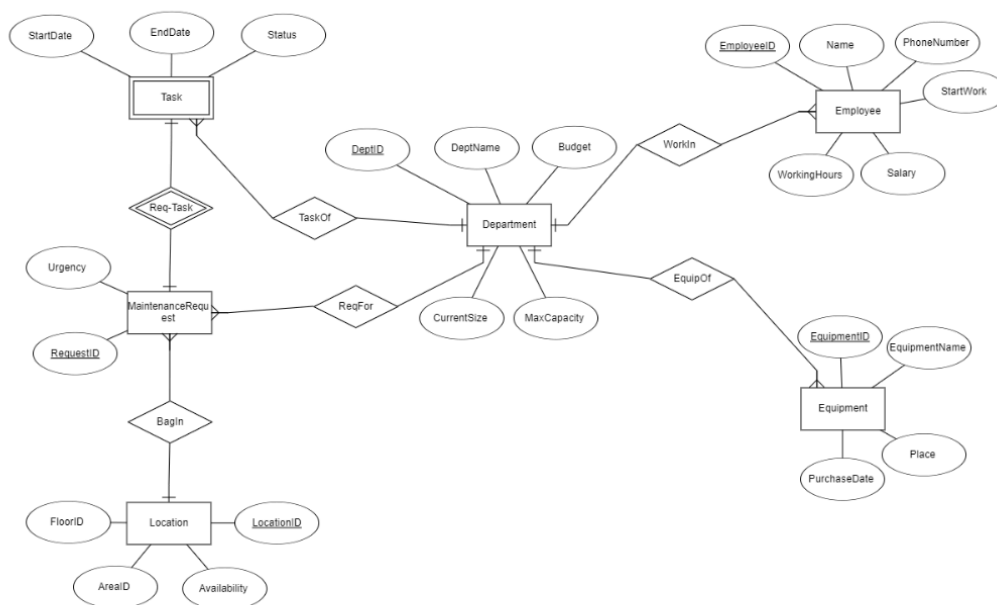
תרשים DSD שלנו:



תרשים ERD שלהם:



תרשים ERD שלנו:



שינויים:

:Employee

מאחדים את em_id מהסכמה שלהם ואת EmployeeID מהסכמה שלנו.

מורידים את first_name וlast_name כפי שיש בסכמה שלהם והופכים אותם לName שלנו.

מאחדים את Salary.

מוסיפים את PhoneNumber, StartWork, WorkingHours מהסכמה שלנו, וזה יראה כך:

Employee(**EmployeeID(em_id)**, Name(first name & last name), PhoneNumber, StartWork, **DeptID(FK)**, Salary, WorkingHours)

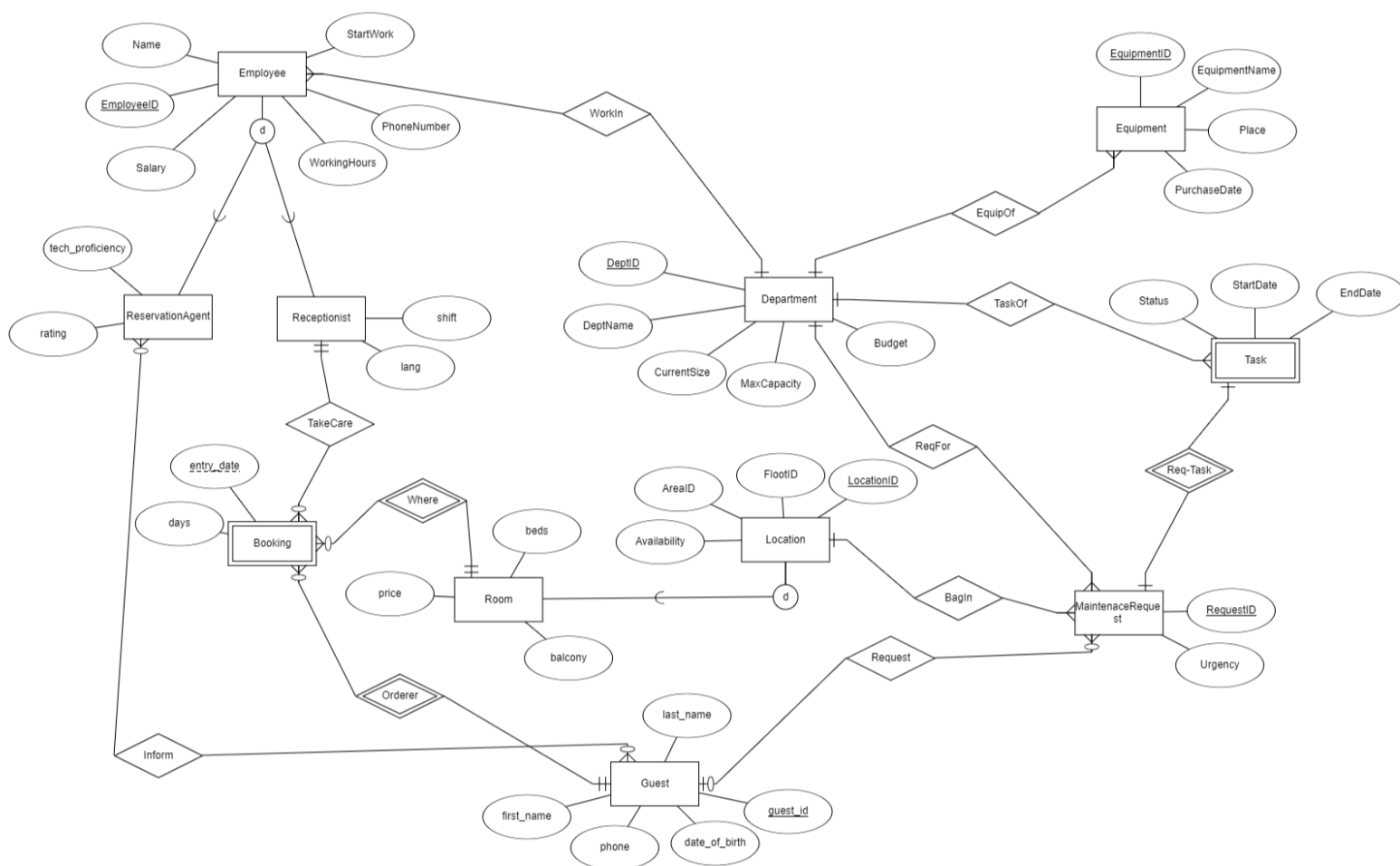
:Room

נשנה את החדר כך שירש מLocationID וזה בעצם אומר לנו שכל חדר אמור להיות סוג של מיקום. לRoom אין יותר room_number אלא יש לו LocationID בגלל הירושה מהמיקום.

:Request הקשר

זהו הקשר שהיה בין Room ל Guest. כעת נשנה אותו כך שיהיה קשר בין MaintenanceRequest ל Guest, כך שלכל בקשת תחזוקה יכול להיות רק אורח אחד ששלח אותה ולכל אורח יכולות להיות כמה בקשות תחזוקה. (לא יראו את הקשר הזה בתרשים הDSD כי קודם הוא היה קשר רבים לרבים אבל עכשיו הוא קשר רבים ליחיד, אז הוא לא מופע בDSD).

תרשים ERD מאוחד:



תרשים DSD מאוחד:

פקודות שינוי ויצירה חדשות:

```

1  -- Drop the Request table if it exists
2  DROP TABLE Request;
3
4  -- Drop constraints from the Booking and ROOM tables
5  ALTER TABLE Booking DROP CONSTRAINT FK_Booking;
6  ALTER TABLE Booking DROP CONSTRAINT FK_BOOKING_ROOM_NUMBER;
7  ALTER TABLE ROOM DROP CONSTRAINT FK_ROOM;
8
9  -- Drop constraints from the Inform, ReservationAgent, and Receptionist tables
10 ALTER TABLE Inform DROP CONSTRAINT FK_INFORM;
11 ALTER TABLE Inform DROP CONSTRAINT FK_EM_ID_INFORM;
12 ALTER TABLE ReservationAgent DROP CONSTRAINT FK_RESERVATIONAGENT_EM_ID;
13 ALTER TABLE ReservationAgent DROP CONSTRAINT FK_RESERVATIONAGENT;
14 ALTER TABLE Receptionist DROP CONSTRAINT FK_RECEPTIONIST_EM_ID;
15 ALTER TABLE Receptionist DROP CONSTRAINT FK_RECEPTIONIST;
16
17
18 -- Step 1: Add a default value constraint to PhoneNumber if not already present
19 ALTER TABLE Employee MODIFY PhoneNumber DEFAULT '000-000-0000';
20
21 -- Step 2: Merge Data from Employee2 into Employee
22 MERGE INTO Employee e
23 USING (
24     SELECT em_id, first_name || ' ' || last_name AS full_name, salary
25     FROM Employee2
26 ) e2
27 ON (e.EmployeeID = e2.em_id)
28 WHEN MATCHED THEN
29     UPDATE SET e.Name = e2.full_name, e.Salary = e2.salary
30 WHEN NOT MATCHED THEN
31     INSERT (EmployeeID, Name, PhoneNumber, StartWork, Salary, WorkingHours, DeptID)
32     VALUES (e2.em_id, e2.full_name, '000-000-0000', SYSDATE, e2.salary, 0, 1); -- Provide a default phone number
33
34 -- Step 3: Drop the Employee2 Table
35 DROP TABLE Employee2;
36
37
38 -- Drop room_number columns from ROOM and Booking tables
39 ALTER TABLE ROOM DROP COLUMN room_number;
40 ALTER TABLE Booking DROP COLUMN room_number;
41
42 -- Add a temporary column to ROOM, Booking, and Location_ tables to store row numbers
43 ALTER TABLE ROOM ADD temp INT;
44 ALTER TABLE Location_ ADD temp INT;
45 ALTER TABLE Booking ADD temp INT;
46
47 -- Update the temporary columns with row numbers
48 UPDATE ROOM SET temp = rownum;
49 UPDATE Location_ SET temp = rownum;
50 UPDATE Booking SET temp = rownum;

```

```

52 -- Add a LocationID column to the ROOM table and set up a foreign key constraint
53 ALTER TABLE ROOM ADD LocationID INT;
54 ALTER TABLE ROOM ADD CONSTRAINT FK_ROOM_LocationID FOREIGN KEY (LocationID) REFERENCES Location_(LocationID);
55
56 DECLARE
57     v_location_count NUMBER;
58     v_updated_count NUMBER := 0;
59 BEGIN
60     -- Get the count of locations
61     SELECT COUNT(*) INTO v_location_count FROM Location_;
62     DBMS_OUTPUT.PUT_LINE('Location count: ' || v_location_count);
63
64     -- Update the LocationID in ROOM table based on temp columns
65     UPDATE ROOM r
66     SET LocationID = (
67         SELECT LocationID
68         FROM Location_ l
69         WHERE l.temp = MOD(r.temp - 1, v_location_count) + 1
70     )
71     WHERE r.LocationID IS NULL;
72
73     v_updated_count := SQL%ROWCOUNT;
74     DBMS_OUTPUT.PUT_LINE('Updated ' || v_updated_count || ' rooms.');
```

הפעל את Windows

עבור אל 'הגדרות' כדי להפעיל את Windows.

```

75
76 -- Add primary key constraint to the ROOM table
77 BEGIN
78     EXECUTE IMMEDIATE 'ALTER TABLE ROOM ADD CONSTRAINT FK_ROOM PRIMARY KEY (LocationID)';
79     DBMS_OUTPUT.PUT_LINE('Primary key constraint added to ROOM table.');
```

הפעל את Windows

עבור אל 'הגדרות' כדי להפעיל את Windows.

```

80 E/CEPTION
81 WHEN OTHERS THEN
82     IF SQLCODE = -2260 THEN -- ORA-02260: table can have only one primary key
83         DBMS_OUTPUT.PUT_LINE('Primary key constraint already exists on ROOM table.');
```

הפעל את Windows

עבור אל 'הגדרות' כדי להפעיל את Windows.

```

84     ELSE
85         RAISE;
86     END IF;
87 END;
88
89 -- Add LocationID column to Booking table
90 BEGIN
91     EXECUTE IMMEDIATE 'ALTER TABLE Booking ADD LocationID INT';
92     DBMS_OUTPUT.PUT_LINE('Column LocationID added successfully to Booking table.');
```

הפעל את Windows

עבור אל 'הגדרות' כדי להפעיל את Windows.

```

93 E/CEPTION
94 WHEN OTHERS THEN
95     IF SQLCODE = -1430 THEN -- ORA-01430: column being added already exists in table
96         DBMS_OUTPUT.PUT_LINE('Column LocationID already exists in Booking table.');
```

הפעל את Windows

עבור אל 'הגדרות' כדי להפעיל את Windows.

```

97     ELSE
98         RAISE;
99     END IF;
100 END;
```

```

102 COMMIT;
103 DEMS_OUTPUT.PUT_LINE('All operations completed successfully.');
```

```

104 E/CEPTION
105 WHEN OTHERS THEN
106     DEMS_OUTPUT.PUT_LINE('An error occurred: ' || SQLERRM);
107     ROLLBACK;
108 END;
109
110 ALTER TABLE Booking ADD CONSTRAINT FK_Booking_LocationID FOREIGN KEY (LocationID) REFERENCES Location_(LocationID);
111 DECLARE
112     v_location_count NUMBER;
113     v_updated_count NUMBER;
114 BEGIN
115     -- Get the count of locations
116     SELECT COUNT(*) INTO v_location_count FROM Location_;
117     DEMS_OUTPUT.PUT_LINE('Location count: ' || v_location_count);
118
119     -- Update the LocationID in Booking table based on temp columns
120     UPDATE Booking b
121     SET b.LocationID = (
122         SELECT l.LocationID
123         FROM Location_ l
124         WHERE l.temp = CASE
125             WHEN MOD(b.temp - 1, v_location_count) = 0 THEN v_location_count
126             ELSE MOD(b.temp - 1, v_location_count)
127         END
128     )
129     WHERE b.LocationID IS NULL; -- Only update rows where LocationID is not set
130
131     v_updated_count := SQL%ROWCOUNT;
132     DEMS_OUTPUT.PUT_LINE('Update completed. ' || v_updated_count || ' bookings updated.');
```

```

133
134 -- Add primary key constraint to the Booking table
135 BEGIN
136     E/ECUTE IMMEDIATE 'ALTER TABLE Booking ADD CONSTRAINT PK_Booking PRIMARY KEY (LocationID, guest_id, entry_date)';
137     DEMS_OUTPUT.PUT_LINE('Primary key constraint added to Booking table.');
```

```

138 E/CEPTION
139 WHEN OTHERS THEN
140     IF SQLCODE = -2260 THEN -- ORA-02260: table can have only one primary key
141         DEMS_OUTPUT.PUT_LINE('Primary key constraint already exists on Booking table.');
```

```

142     ELSE
143         RAISE;
144     END IF;
145 END;
146
147 COMMIT;
```

```

148 DEMS_OUTPUT.PUT_LINE('All operations completed successfully.');
```

```

149 EXCEPTION
150 WHEN OTHERS THEN
151     DEMS_OUTPUT.PUT_LINE('An error occurred: ' || SQLERRM);
152     ROLLBACK;
153 END;
154
155
156 -- Drop the em_id columns from ReservationAgent, Receptionist, Booking, and Inform tables
157 ALTER TABLE ReservationAgent DROP COLUMN em_id;
158 ALTER TABLE Receptionist DROP COLUMN em_id;
159 ALTER TABLE Booking DROP COLUMN em_id;
160 ALTER TABLE Inform DROP COLUMN em_id;
161
162 -- Add a temporary column to EmployeeID, ReservationAgent, Receptionist, and Inform tables to store row numbers
163 ALTER TABLE Employee ADD temp INT;
164 ALTER TABLE ReservationAgent ADD temp INT;
165 ALTER TABLE Receptionist ADD temp INT;
166 ALTER TABLE Inform ADD temp INT;
167
168 -- Update the temporary columns with row numbers
169 UPDATE Employee SET temp = rownum;
170 UPDATE ReservationAgent SET temp = rownum;
171 UPDATE Receptionist SET temp = rownum;
172 UPDATE Inform SET temp = rownum;
173
174 -- Add an EmployeeID column to ReservationAgent table and set up a foreign key constraint
175 ALTER TABLE ReservationAgent ADD EmployeeID INT;
176 ALTER TABLE ReservationAgent ADD CONSTRAINT FK_ReservationAgent_EmployeeID FOREIGN KEY (EmployeeID) REFERENCES EMPLOYEE(EmployeeID);
177
178 DECLARE
179     v_employee_count NUMBER;
180     v_reservation_agent_count NUMBER;
181     v_count NUMBER;
182     v_updated_count NUMBER := 0;
183 BEGIN
184     -- Get the count of employees
185     SELECT COUNT(*) INTO v_employee_count FROM Employee;
186     DEMS_OUTPUT.PUT_LINE('Employee count: ' || v_employee_count);
187
188     -- Get the count of reservation agents
189     SELECT COUNT(*) INTO v_reservation_agent_count FROM ReservationAgent;
190     DEMS_OUTPUT.PUT_LINE('ReservationAgent count: ' || v_reservation_agent_count);
191
192     -- Determine which count to use for the modulo operation
193     v_count := LEAST(v_employee_count, v_reservation_agent_count);
194     DEMS_OUTPUT.PUT_LINE('Count used for modulo: ' || v_count);

```

```

196 -- Update the EmployeeID in ReservationAgent table
197 UPDATE ReservationAgent ra
198 SET EmployeeID = (
199     SELECT e.EmployeeID
200     FROM Employee e
201     WHERE e.temp = CASE
202         WHEN MOD(ra.temp - 1, v_count) = 0 THEN v_count
203         ELSE MOD(ra.temp - 1, v_count)
204     END
205 )
206 WHERE ra.EmployeeID IS NULL;
207
208 v_updated_count := SQL%ROWCOUNT;
209 DEMS_OUTPUT.PUT_LINE('Update completed. ' || v_updated_count || ' reservation agents updated.');
```

```

210
211 -- Add primary key constraint to the ReservationAgent table
212 BEGIN
213     E/ECUTE IMMEDIATE 'ALTER TABLE ReservationAgent ADD CONSTRAINT PK_ReservationAgent PRIMARY KEY (EmployeeID)';
214     DEMS_OUTPUT.PUT_LINE('Primary key constraint added to ReservationAgent table.');
```

```

215 E/CEPTION
216 WHEN OTHERS THEN
217     IF SQLCODE = -2260 THEN -- ORA-02260: table can have only one primary key
218         DEMS_OUTPUT.PUT_LINE('Primary key constraint already exists on ReservationAgent table.');
```

```

219     ELSE
220         RAISE;
221     END IF;
222 END;
223
224 -- Add an EmployeeID column to Receptionist table
225 BEGIN
226     E/ECUTE IMMEDIATE 'ALTER TABLE Receptionist ADD EmployeeID INT';
227     DEMS_OUTPUT.PUT_LINE('EmployeeID column added to Receptionist table.');
```

```

228 E/CEPTION
229 WHEN OTHERS THEN
230     IF SQLCODE = -1430 THEN -- ORA-01430: column being added already exists in table
231         DEMS_OUTPUT.PUT_LINE('EmployeeID column already exists in Receptionist table.');
```

```

232     ELSE
233         RAISE;
234     END IF;
235 END;

```

```

237 -- Set up a foreign key constraint on Receptionist table
238 BEGIN
239     E/ECUTE IMMEDIATE 'ALTER TABLE Receptionist ADD CONSTRAINT Receptionist_EmployeeID FOREIGN KEY (EmployeeID) REFERENCES EMPLOYEE(EmployeeID)';
240     DEMS_OUTPUT.PUT_LINE('Foreign key constraint added to Receptionist table.');
```

```

241 E/CEPTION
242 WHEN OTHERS THEN
243     IF SQLCODE = -2275 THEN -- ORA-02275: such a referential constraint already exists in the table
244         DEMS_OUTPUT.PUT_LINE('Foreign key constraint already exists on Receptionist table.');
```

```

245     ELSE
246         RAISE;
247     END IF;
248 END;
249
250 COMMIT;
251 DEMS_OUTPUT.PUT_LINE('All operations completed successfully.');
```

```

252 E/CEPTION
253 WHEN OTHERS THEN
254     DEMS_OUTPUT.PUT_LINE('An error occurred: ' || SQLERRM);
255     ROLLBACK;
256 END;
257
258
259 DECLARE
260     v_employee_count NUMBER;
261     v_receptionist_count NUMBER;
262     v_count NUMBER;
263     v_updated_count NUMBER := 0;
264 BEGIN
265     -- Get the count of employees
266     SELECT COUNT(*) INTO v_employee_count FROM Employee;
267     DEMS_OUTPUT.PUT_LINE('Employee count: ' || v_employee_count);
268
269     -- Get the count of receptionists
270     SELECT COUNT(*) INTO v_receptionist_count FROM Receptionist;
271     DEMS_OUTPUT.PUT_LINE('Receptionist count: ' || v_receptionist_count);
272
273     -- Determine which count to use for the modulo operation
274     v_count := LEAST(v_employee_count, v_receptionist_count);
275     DEMS_OUTPUT.PUT_LINE('Count used for modulo: ' || v_count);
276
277     -- Update the EmployeeID in Receptionist table
278     UPDATE Receptionist r
279     SET EmployeeID = (
280         SELECT e.EmployeeID
281         FROM Employee e
282         WHERE e.temp = CASE
283             WHEN MOD(r.temp - 1, v_count) = 0 THEN v_count
284             ELSE MOD(r.temp - 1, v_count)
285         END
286     )

```



```

287 WHERE r.EmployeeID IS NULL;
288 v_updated_count := SQL%ROWCOUNT;
289 DEMS_OUTPUT.PUT_LINE('Update completed. ' || v_updated_count || ' receptionists updated.');
```

-- Add primary key constraint to the Receptionist table

```

291 BEGIN
292     />ECUTE IMMEDIATE 'ALTER TABLE Receptionist ADD CONSTRAINT PK_Receptionist PRIMARY KEY (EmployeeID)';
293     DEMS_OUTPUT.PUT_LINE('Primary key constraint added to Receptionist table.');
```

E/CEPTION

```

295     WHEN OTHERS THEN
296         IF SQLCODE = -2260 THEN -- ORA-02260: table can have only one primary key
297             DEMS_OUTPUT.PUT_LINE('Primary key constraint already exists on Receptionist table.');
```

ELSE

```

299             DEMS_OUTPUT.PUT_LINE('Failed to add primary key constraint to Receptionist table: ' || SQLERRM);
300         END IF;
301     END;
```

-- Add an EmployeeID column to Booking table

```

303 BEGIN
304     />ECUTE IMMEDIATE 'ALTER TABLE Booking ADD EmployeeID INT';
305     DEMS_OUTPUT.PUT_LINE('EmployeeID column added to Booking table.');
```

E/CEPTION

```

307     WHEN OTHERS THEN
308         IF SQLCODE = -1430 THEN -- ORA-01430: column being added already exists in table
309             DEMS_OUTPUT.PUT_LINE('EmployeeID column already exists in Booking table.');
```

ELSE

```

311             DEMS_OUTPUT.PUT_LINE('Failed to add EmployeeID column to Booking table: ' || SQLERRM);
312         END IF;
313     END;
```

-- Set up a foreign key constraint on Booking table

```

315 BEGIN
316     />ECUTE IMMEDIATE 'ALTER TABLE Booking ADD CONSTRAINT FK_Booking_EmployeeID FOREIGN KEY (EmployeeID) REFERENCES EMPLOYEE(EmployeeID)';
317     DEMS_OUTPUT.PUT_LINE('Foreign key constraint added to Booking table.');
```

E/CEPTION

```

319     WHEN OTHERS THEN
320         IF SQLCODE = -2275 THEN -- ORA-02275: such a referential constraint already exists in the table
321             DEMS_OUTPUT.PUT_LINE('Foreign key constraint already exists on Booking table.');
```

ELSE

```

323             DEMS_OUTPUT.PUT_LINE('Failed to add foreign key constraint to Booking table: ' || SQLERRM);
324         END IF;
325     END;
```

COMMIT;

```

327 DEMS_OUTPUT.PUT_LINE('All operations completed successfully.');
```

E/CEPTION

```

329     WHEN OTHERS THEN
330         DEMS_OUTPUT.PUT_LINE('An error occurred during the process: ' || SQLERRM);
331     ROLLBACK;
```

END;

הפעולות ב-Windows
עבור אל 'הגדרות' כדי להפעיל את Windows

```

339 DECLARE
340     v_employee_count NUMBER;
341     v_booking_count NUMBER;
342     v_count NUMBER;
343     v_updated_count NUMBER := 0;
344 BEGIN
345     -- Get the count of employees
346     SELECT COUNT(*) INTO v_employee_count FROM Employee;
347     DEMS_OUTPUT.PUT_LINE('Employee count: ' || v_employee_count);
348
349     -- Get the count of bookings
350     SELECT COUNT(*) INTO v_booking_count FROM Booking;
351     DEMS_OUTPUT.PUT_LINE('Booking count: ' || v_booking_count);
352
353     -- Determine which count to use for the modulo operation
354     v_count := LEAST(v_employee_count, v_booking_count);
355     DEMS_OUTPUT.PUT_LINE('Count used for modulo: ' || v_count);
356
357     -- Update the EmployeeID in Booking table
358     UPDATE Booking b
359     SET EmployeeID = (
360         SELECT e.EmployeeID
361         FROM Employee e
362         WHERE e.temp = CASE
363             WHEN MOD(b.temp - 1, v_count) = 0 THEN v_count
364             ELSE MOD(b.temp - 1, v_count)
365         END
366     )
367     WHERE b.EmployeeID IS NULL;
368
369     v_updated_count := SQL%ROWCOUNT;
370     DEMS_OUTPUT.PUT_LINE('Update completed. ' || v_updated_count || ' bookings updated.');
```

-- Add an EmployeeID column to Inform table

```

372 BEGIN
373     E/ECUTE IMMEDIATE 'ALTER TABLE Inform ADD EmployeeID INT';
374     DEMS_OUTPUT.PUT_LINE('EmployeeID column added to Inform table.');
```

E/CEPTION

```

377 WHEN OTHERS THEN
378     IF SQLCODE = -1430 THEN -- ORA-01430: column being added already exists in table
379         DEMS_OUTPUT.PUT_LINE('EmployeeID column already exists in Inform table.');
```

ELSE

```

381         DEMS_OUTPUT.PUT_LINE('Failed to add EmployeeID column to Inform table: ' || SQLERRM);
382     END IF;
383 END;
```

```

385 -- Set up a foreign key constraint on Inform table
386 BEGIN
387     EXECUTE IMMEDIATE 'ALTER TABLE Inform ADD CONSTRAINT FK_Inform_EmployeeID FOREIGN KEY (EmployeeID) REFERENCES EMPLOYEE(EmployeeID)';
388     DEMS_OUTPUT.PUT_LINE('Foreign key constraint added to Inform table. ');
389 E/CEPTION
390     WHEN OTHERS THEN
391         IF SQLCODE = -2275 THEN -- ORA-02275: such a referential constraint already exists in the table
392             DEMS_OUTPUT.PUT_LINE('Foreign key constraint already exists on Inform table. ');
393         ELSE
394             DEMS_OUTPUT.PUT_LINE('Failed to add foreign key constraint to Inform table: ' || SQLERRM);
395         END IF;
396     END;
397
398 COMMIT;
399 DEMS_OUTPUT.PUT_LINE('All operations completed successfully. ');
400 E/CEPTION
401     WHEN OTHERS THEN
402         DEMS_OUTPUT.PUT_LINE('An error occurred during the process: ' || SQLERRM);
403         ROLLBACK;
404 END;
405
406 DECLARE
407     v_employee_count NUMBER;
408     v_inform_count NUMBER;
409     v_count NUMBER;
410     v_updated_count NUMBER := 0;
411 BEGIN
412     -- Get the count of employees
413     SELECT COUNT(*) INTO v_employee_count FROM Employee;
414     DEMS_OUTPUT.PUT_LINE('Employee count: ' || v_employee_count);
415
416     -- Get the count of inform records
417     SELECT COUNT(*) INTO v_inform_count FROM Inform;
418     DEMS_OUTPUT.PUT_LINE('Inform record count: ' || v_inform_count);
419
420     -- Determine which count to use for the modulo operation
421     v_count := LEAST(v_employee_count, v_inform_count);
422     DEMS_OUTPUT.PUT_LINE('Count used for modulo: ' || v_count);
423
424     -- Update the EmployeeID in Inform table
425     UPDATE Inform i
426     SET EmployeeID = (
427         SELECT e.EmployeeID
428         FROM Employee e
429         WHERE e.temp = CASE
430             WHEN MOD(i.temp - 1, v_count) = 0 THEN v_count
431             ELSE MOD(i.temp - 1, v_count)
432         END
433     )
434     WHERE i.EmployeeID IS NULL;

```

```

436 v_updated_count := SQL%ROWCOUNT;
437 DEMS_OUTPUT.PUT_LINE('Update completed. ' || v_updated_count || ' inform records updated. ');
438
439 -- Add primary key constraint to the Inform table
440 BEGIN
441     E/ECUTE IMMEDIATE 'ALTER TABLE Inform ADD CONSTRAINT PK_Inform PRIMARY KEY (EmployeeID)';
442     DEMS_OUTPUT.PUT_LINE('Primary key constraint added to Inform table. ');
443 E/CEPTION
444     WHEN OTHERS THEN
445         IF SQLCODE = -2260 THEN -- ORA-02260: table can have only one primary key
446             DEMS_OUTPUT.PUT_LINE('Primary key constraint already exists on Inform table. ');
447         ELSE
448             DEMS_OUTPUT.PUT_LINE('Failed to add primary key constraint to Inform table: ' || SQLERRM);
449         END IF;
450 END;
451
452 -- Add a temporary column to MaintenanceRequest_ table
453 BEGIN
454     E/ECUTE IMMEDIATE 'ALTER TABLE MaintenanceRequest_ ADD temp INT';
455     DEMS_OUTPUT.PUT_LINE('Temporary column added to MaintenanceRequest_ table. ');
456 E/CEPTION
457     WHEN OTHERS THEN
458         IF SQLCODE = -1430 THEN -- ORA-01430: column being added already exists in table
459             DEMS_OUTPUT.PUT_LINE('Temporary column already exists in MaintenanceRequest_ table. ');
460         ELSE
461             DEMS_OUTPUT.PUT_LINE('Failed to add temporary column to MaintenanceRequest_ table: ' || SQLERRM);
462         END IF;
463 END;
464
465 -- Add a temporary column to Guest table
466 BEGIN
467     E/ECUTE IMMEDIATE 'ALTER TABLE Guest ADD temp INT';
468     DEMS_OUTPUT.PUT_LINE('Temporary column added to Guest table. ');
469 E/CEPTION
470     WHEN OTHERS THEN
471         IF SQLCODE = -1430 THEN -- ORA-01430: column being added already exists in table
472             DEMS_OUTPUT.PUT_LINE('Temporary column already exists in Guest table. ');
473         ELSE
474             DEMS_OUTPUT.PUT_LINE('Failed to add temporary column to Guest table: ' || SQLERRM);
475         END IF;
476 END;
477
478 COMMIT;
479 DEMS_OUTPUT.PUT_LINE('All operations completed successfully. ');
480 E/CEPTION
481     WHEN OTHERS THEN
482         DEMS_OUTPUT.PUT_LINE('An error occurred during the process: ' || SQLERRM);
483         ROLLBACK;
484 END;

```

```

487 -- Update the temporary columns with row numbers
488 UPDATE MaintenanceRequest_ SET temp = rownum;
489 UPDATE Guest SET temp = rownum;
490
491 -- Add a guest_id column to MaintenanceRequest_ table and set up a foreign key constraint
492 ALTER TABLE MaintenanceRequest_ ADD guest_id INT;
493 ALTER TABLE MaintenanceRequest_ ADD CONSTRAINT FK_MaintenanceRequest_Guest FOREIGN KEY (guest_id) REFERENCES Guest(guest_id);
494
495 DECLARE
496     v_guest_count NUMBER;
497     v_maintenance_request_count NUMBER;
498     v_count NUMBER;
499 BEGIN
500     -- Get the count of guests
501     SELECT COUNT(*) INTO v_guest_count FROM Guest;
502
503     -- Get the count of maintenance requests
504     SELECT COUNT(*) INTO v_maintenance_request_count FROM MaintenanceRequest_;
505
506     -- Determine which count to use for the modulo operation
507     v_count := LEAST(v_guest_count, v_maintenance_request_count);
508
509     -- Update the guest_id in MaintenanceRequest_ table
510     FOR mr IN (SELECT temp FROM MaintenanceRequest_)
511     LOOP
512         UPDATE MaintenanceRequest_
513         SET guest_id = (
514             SELECT g.guest_id
515             FROM Guest g
516             WHERE g.temp = MOD(mr.temp - 1, v_count) + 1
517         )
518         WHERE temp = mr.temp;
519     END LOOP;
520
521 COMMIT;
522 E/CEPTION
523 WHEN OTHERS THEN
524     DEMS_OUTPUT.PUT_LINE('An error occurred: ' || SQLERRM);
525     ROLLBACK;
526 END;

```

מה בעצם עשינו בקוד הנ"ל:

בקוד הנ"ל אנו מבצעים את השאילתות הבאות:

מוחקים את הטבלה "Request" אם היא קיימת.

מוחקים אילוצים (constraints) מטבלאות "Booking" ו-"ROOM".

מוחקים אילוצים מטבלאות "ReservationAgent", "Inform" ו-"Receptionist".

מוסיפים ערך ברירת מחדל לשדה "PhoneNumber" בטבלה "Employee".

ממזגים נתונים מטבלת "Employee2" לתוך טבלת "Employee".

מוחקים את טבלת "Employee2".

מוחקים את עמודות "room_number" מטבלאות "ROOM" ו-"Booking".

מוסיפים עמודות זמניות לטבלאות "Location", "_ROOM", ו-"Booking" כדי לאחסן מספרי שורות.

מעדכנים את עמודות השורות הזמניות עם מספרי השורות.

מוסיפים עמודת "LocationID" לטבלת "ROOM" ומגדירים אילוץ מפתח זר.

מעדכנים את עמודת "LocationID" בטבלת "ROOM" בהתבסס על מספרי השורות הזמניים.

מוסיפים אילוץ מפתח ראשי לטבלת "ROOM".

מוסיפים עמודת "LocationID" לטבלת "Booking" ומגדירים אילוץ מפתח זר.

מעדכנים את עמודת "LocationID" בטבלת "Booking" בהתבסס על מספרי השורות הזמניים.

מוסיפים אילוץ מפתח ראשי לטבלת "Booking".

מוחקים עמודות "em_id" מטבלאות "ReservationAgent", "Receptionist", ו-"Inform".

מוסיפים עמודות זמניות לטבלאות "ReservationAgent", "Receptionist", ו-"Employee".

ו-"Inform" כדי לאחסן מספרי שורות.

מעדכנים את עמודות השורות הזמניות עם מספרי השורות.

מוסיפים עמודת "EmployeeID" לטבלת "ReservationAgent" ומגדירים אילוץ מפתח זר.

מעדכנים את עמודת "EmployeeID" בטבלת "ReservationAgent" בהתבסס על מספרי השורות הזמניים ומוסיפים אילוץ מפתח ראשי.

מוסיפים עמודת "EmployeeID" לטבלת "Receptionist" ומגדירים אילוץ מפתח זר.

מעדכנים את עמודת "EmployeeID" בטבלת "Receptionist" בהתבסס על מספרי השורות הזמניים ומוסיפים אילוץ מפתח ראשי.

מוסיפים עמודת "EmployeeID" לטבלת "Booking" ומגדירים אילוץ מפתח זר.

מעדכנים את עמודת "EmployeeID" בטבלת "Booking" בהתבסס על מספרי השורות הזמניים.

מוסיפים עמודת "EmployeeID" לטבלת "Inform" ומגדירים אילוץ מפתח זר.

מעדכנים את עמודת "EmployeeID" בטבלת "Inform" בהתבסס על מספרי השורות הזמניים ומוסיפים אילוץ מפתח ראשי.

מוסיפים עמודות זמניות לטבלאות "_MaintenanceRequest" ו-"Guest" כדי לאחסן מספרי שורות.

מעדכנים את עמודות השורות הזמניות עם מספרי השורות.

מוסיפים עמודת "guest_id" לטבלת "_MaintenanceRequest" ומגדירים אילוץ מפתח זר.

מעדכנים את עמודת "guest_id" בטבלת "_MaintenanceRequest" בהתבסס על מספרי השורות הזמניים.

המבטים(VIEWS):

יצירת EmployeeDetails:

```
CREATE VIEW EmployeeDetails AS
SELECT
    e.EmployeeID,
    e.Name,
    e.Salary,
    e.PhoneNumber,
    e.StartWork,
    e.WorkingHours,
    d.DeptName,
    d.Budget,
    d.MaxCapacity,
    d.CurrentSize,
    r.shift AS ReceptionistShift,
    r.lang AS ReceptionistLanguage,
    ra.tech_proficiency AS ReservationAgentTechProficiency,
    ra.rating AS ReservationAgentRating
FROM
    Employee e
JOIN
    Department d ON e.DeptID = d.DeptID
LEFT JOIN
    Receptionist r ON e.EmployeeID = r.EmployeeID
LEFT JOIN
    ReservationAgent ra ON e.EmployeeID = ra.EmployeeID;
```

תיאור: המבט יוצר תצוגה משולבת של פרטי עובדים, הכוללת מידע מטבלת העובדים, פרטי מחלקה מטבלת המחלקות, ונתונים נוספים במידה והעובד הוא פקיד קבלה (Receptionist) או סוכן הזמנות (ReservationAgent).

תיאור שאילתא:

השאילתא מציגה את תעודת הזהות של העובד, השם של העובד, שם המחלקה והמשכורת של כל עובד מתוך המבט EmployeeDetails.

```
SELECT
    EmployeeID,
    Name,
    DeptName,
    Salary
FROM
    EmployeeDetails;
```

		EMPLOYEEID	NAME		DEPTNAME		SALARY
▶	1	1469	Wainwright Kiln	***	Security Systems	***	64098
	2	1470	Holly Cray	***	Plumbing	***	46714
	3	1471	Lamont Birks	***	Security Systems	***	84400
	4	1472	Lian Penning	***	Appliance Repair	***	72114
	5	1473	Nicolette Codron	***	Plumbing	***	13878
	6	1474	Mariana Aucoate	***	Security Systems	***	62162
	7	1475	Ardelle Tocqueville	***	Landscaping	***	65773
	8	1476	Jard Beecheno	***	Electrical	***	25278
	9	1477	Valaria Pioli	***	Carpentry	***	18990
	10	1478	Gorden Overland	***	Electrical	***	10122

תיאור השאילתא:

שאילתא זו מציגה את תעודת הזהות של העובד, השם של העובד, משמרת פקיד הקבלה ושפת פקיד הקבלה לכל העובדים שהם פקידי קבלה מתוך המבט, EmployeeDetails.

<pre> SELECT EmployeeID, Name, ReceptionistShift, ReceptionistLanguage FROM EmployeeDetails WHERE ReceptionistShift IS NOT NULL; </pre>						
		EMPLOYEEID	NAME	RECEPTIONISTSHIFT	RECEPTIONISTLANGUAGE	
▶	1	1852	Kassandra Treleven	*** Evening	Bosnian	***
	2	1469	Wainwright Kiln	*** Night	Mã,Âori	***
	3	1470	Holly Cray	*** Morning	Tok Pisin	***
	4	1471	Lamont Birks	*** Morning	Kashmiri	***
	5	1472	Lian Penning	*** Night	Tamil	***
	6	1473	Nicolette Codron	*** Evening	Hebrew	***
	7	1474	Mariana Aucoate	*** Morning	Albanian	***
	8	1475	Ardelle Tocqueville	*** Morning	Swahili	***
	9	1476	Jard Beecheno	*** Morning	Punjabi	***
	10	1477	Valaria Pioli	*** Night	Ndebele	***

יצירת GuestDetails:

תיאור המבט: המבט יוצר תצוגה משולבת של פרטי אורחים, הכוללת מידע מטבלת האורחים, פרטי הזמנה מטבלת ההזמנות, ומידע על מיקום מתוך הטבלת המיקום.

```
CREATE VIEW GuestDetails AS
SELECT
    g.guest_id,
    g.first_name,
    g.last_name,
    g.phone,
    g.date_of_birth,
    b.days AS BookingDays,
    b.entry_date AS BookingEntryDate,
    b.LocationID AS BookingLocationID,
    l.FloorID,
    l.AreaID,
    l.Availability
FROM
    Guest g
LEFT JOIN
    Booking b ON g.guest_id = b.guest_id
LEFT JOIN
    Location_1 l ON b.LocationID = l.LocationID;
```

תיאור שאילתא:

השאילתא מציגה את תעודת הזהות של האורח, שם פרטי, שם משפחה, תאריך הכניסה להזמנה, מספר הימים של ההזמנה ומזהה האזור לכל האורחים מתוך המבט GuestDetails.

```
SELECT
    guest_id,
    first_name,
    last_name,
    BookingEntryDate,
    BookingDays,
    AreaID
FROM
    GuestDetails;
```

		GUEST_ID	FIRST_NAME	LAST_NAME	BOOKINGENTRYDATE	BOOKINGDAYS	AREAID
▶	1	954302901	Theobald	Parsell	06/06/1975	14	G606
	2	236336194	Bartolomeo	Juliff	28/12/1970	10	B300
	3	492461632	Noellyn	Kundt	08/10/2004	12	H140
	4	89349836	Noel	Chafer	25/05/1978	1	B009
	5	412264478	Mickey	Dwane	17/09/2000	5	S089
	6	500428309	Kizzee	Tedder	09/04/2000	5	E625
	7	112727921	Aylmar	Cherryman	29/10/2013	13	P242
	8	611577581	Mark	Howgill	10/12/1992	2	I071
	9	774143138	Estella	Seear	05/02/1973	2	L462
	10	789997163	Clemente	Flham	20/10/1975	6	K785

תיאור השאילתא:

השאילתא מציגה את תעודת הזהות של האורח, שם פרטי, שם משפחה ומספר הטלפון של כל האורחים שאין להם תאריך כניסה להזמנה, מתוך המבט GuestDetails.

```
SELECT
```

```
    guest_id,  
    first_name,  
    last_name,  
    phone
```

```
FROM
```

```
    GuestDetails
```

```
WHERE
```

```
    BookingEntryDate IS NULL;
```

		GUEST_ID	FIRST_NAME		LAST_NAME		PHONE
▶	1	825225057	Olin	***	Livick	***	057-4726687
	2	433070647	Danielle	***	Howroyd	***	054-1668113
	3	196104946	Crin	***	Falcus	***	055-4034549
	4	358150371	Guevere	***	Bodiam	***	052-0751006
	5	904309625	Tiffany	***	Seekings	***	052-5653765
	6	787824850	Zacharia	***	Perott	***	059-4017215
	7	770455507	Terza	***	Rahlof	***	058-9713260
	8	868754386	Maitilde	***	Alban	***	054-1713237
	9	846484884	Brewer	***	Manion	***	054-7467956
	10	437808690	Sile	***	Reniefield	***	054-9547493

באמת יש נתונים בכל אחת מהטבלאות.