שערי צדק

ניהול מחלקה

נעם מנדלבאום

תוכן עניינים

שלב א'

מבוא

מערכת הניהול של בית החולים "שערי צדק" כוללת מספר ישויות מרכזיות: רופאים, מטופלים, מחלקות וקשרי רופאים-מטופלים. המערכת נועדה לנהל את כל המידע הרלוונטי אודות המטופלים, הרופאים והמחלקות, על מנת לאפשר ניהול יעיל ומקצועי של כל המידע הנדרש לפעילות תקינה של בית החולים.

ישויות

1. רופאים (Doctors):

טבלת הרופאים תכיל את כל הרופאים העובדים בבית החולים.

- DoctorID (PK) – מספר זיהוי של הרופא.
- FirstName – שם פרטי של הרופא.
- LastName – שם משפחה של הרופא.
- Specialty – מומחיות הרופא.
- Phone – מספר טלפון של הרופא.
- EmailDateOfBirth – תאריך לידה של הרופא.
- HireDate – תאריך העסקה של הרופא.
- Salary – משכורת הרופא.
- DistanceFromHospital – מרחק מגורים מבית החולים.
- DepartmentID (FK) – מזהה המחלקה אליה משתייך הרופא.

2. מטופלים (Patients):

טבלת המטופלים תכיל את כל המטופלים המאושפזים בבית החולים.

- PatientID (PK) – מספר זיהוי של המטופל.
- FirstName – שם פרטי של המטופל.
- LastName – שם משפחה של המטופל.
- Phone – מספר טלפון של המטופל.
- Address – כתובת המגורים של המטופל.
- DateOfBirth – תאריך לידה של המטופל.
- Gender – מגדר המטופל.
- AdmissionDate – תאריך קבלת המטופל.
- ReleaseDate – תאריך שחרור המטופל.
- DepartmentID (FK) – מזהה המחלקה בה מאושפז המטופל.

3. מחלקות (Departments):

טבלת המחלקות תכיל את כל המחלקות בבית החולים.

- DepartmentID (PK) – מספר זיהוי של המחלקה.
- DepartmentName – שם המחלקה.
- BuildingName – שם הבניין בו נמצאת המחלקה.
- Floor – מספר הקומה בה נמצאת המחלקה.
- Phone – מספר טלפון של המחלקה.
- TotalBeds – מספר מיטות כולל במחלקה.
- OccupiedBeds – מספר מיטות בשימוש כרגע.
- HeadOfDepartment – שם ראש המחלקה.

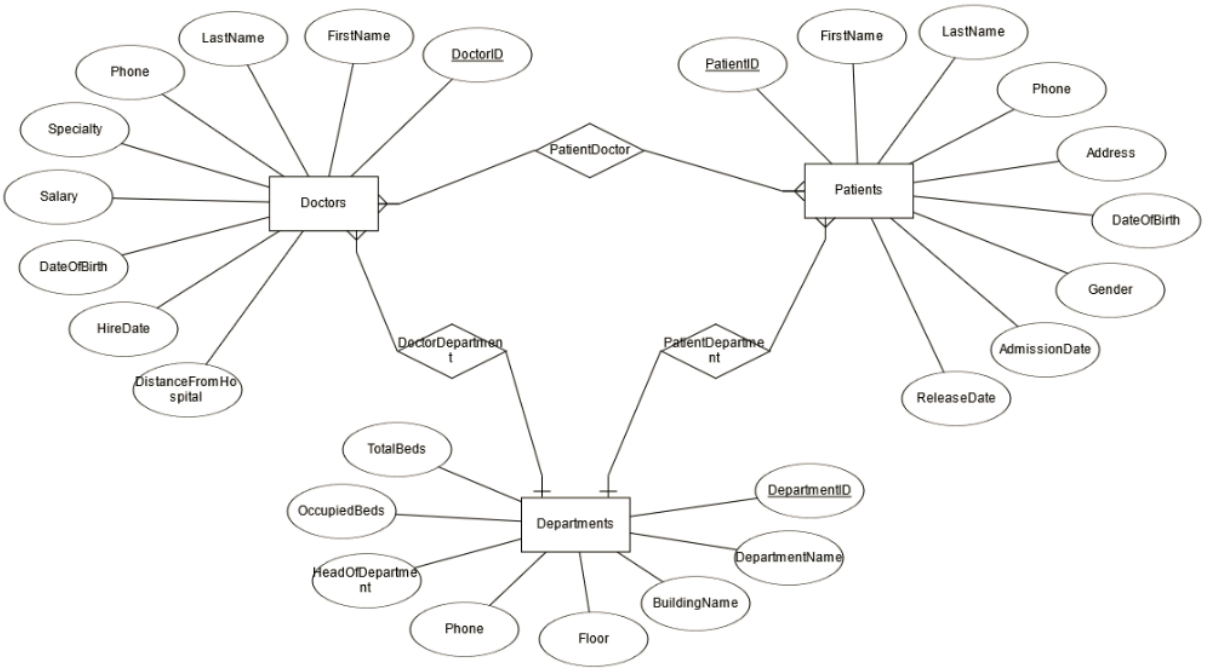
קשרים בין הישויות

PatientDoctor:

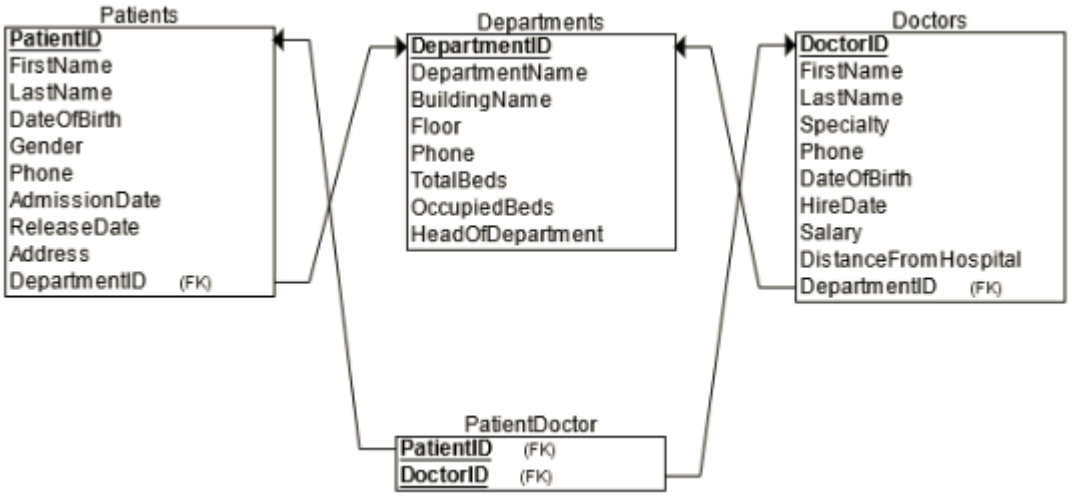
טבלת קשרים המאפשרת את ניהול הקשר בין רופאים למטופלים.

- PatientID (FK) – מספר זיהוי של המטופל.
- DoctorID (FK) – מספר זיהוי של הרופא.

תרשים ERD



תרשים DSD



עמידה ב-3NF

הטבלאות בפרויקט זה עוצבו כך שהן עומדות בדרישות של נרמול לרמה של 3NF, להלן הסיבות לכך:

1. **נרמול לרמה 1NF**:
   - כל הטבלאות מכילות ערכים אטומיים בלבד, כלומר, כל עמודה בטבלה מכילה ערכים יחידים (לא מורכבים ולא חוזרים).
   - אין שורות כפולות בטבלאות, וכל שורה מזוהה באופן ייחודי על ידי מפתח ראשי.
2. **נרמול לרמה 2NF**:
   - כל הטבלאות עומדות בדרישות של 1NF.
   - כל המאפיינים הלא-מפתחותיים בטבלאות תלויים באופן מלא במפתח הראשי, ואין תלות חלקית במפתח הראשי. זה אומר שכל עמודה שאינה חלק מהמפתח הראשי תלויה פונקציונלית בכל המפתח הראשי ולא בחלק ממנו.
3. **נרמול לרמה 3NF**:
   - כל הטבלאות עומדות בדרישות של 2NF.
   - כל המאפיינים הלא-מפתחותיים תלויים ישירות במפתח הראשי בלבד ואינם תלויים במאפיינים לא-מפתחותיים אחרים. אין תלות טרנזיטיבית בין המאפיינים הלא-מפתחותיים בטבלאות.

יצירת הטבלאות

```
CREATE TABLE Departments (
    DepartmentID INT NOT NULL,
    DepartmentName VARCHAR2(15) NOT NULL,
    BuildingName VARCHAR2(15) NOT NULL,
    Floor INT NOT NULL,
    Phone VARCHAR2(13) NOT NULL,
    TotalBeds INT,
    OccupiedBeds INT,
    HeadOfDepartment VARCHAR2(15) NOT NULL,
    PRIMARY KEY (DepartmentID)
);
```

	COLUMN_NAME	DATA_TYPE	DATA_LENGTH	NULLABLE
1	DEPARTMENTID	NUMBER	22	N
2	DEPARTMENTNAME	VARCHAR2	15	N
3	BUILDINGNAME	VARCHAR2	15	N
4	FLOOR	NUMBER	22	N
5	PHONE	VARCHAR2	13	N
6	TOTALBEDS	NUMBER	22	Y
7	OCCUPIEDBEDS	NUMBER	22	Y
8	HEADOFDEPARTMENT	VARCHAR2	15	N

```
CREATE TABLE Doctors
(
  DoctorID INT NOT NULL,
  FirstName VARCHAR(15) NOT NULL,
  LastName VARCHAR(15) NOT NULL,
  Specialty VARCHAR(15) NOT NULL,
  Phone VARCHAR(13) NOT NULL,
  DateOfBirth DATE NOT NULL,
  HireDate DATE NOT NULL,
  Salary INT NOT NULL,
  DistanceFromHospital FLOAT NOT NULL,
  DepartmentID INT NOT NULL,
  PRIMARY KEY (DoctorID),
  FOREIGN KEY (DepartmentID) REFERENCES Departments(DepartmentID)
);
```

	COLUMN_NAME	DATA_TYPE	DATA_LENGTH	NULLABLE
1	DOCTORID	NUMBER	22	N
2	FIRSTNAME	VARCHAR2	15	N
3	LASTNAME	VARCHAR2	15	N
4	SPECIALTY	VARCHAR2	15	N
5	PHONE	VARCHAR2	13	N
6	DATEOFBIRTH	DATE	7	N
7	HIREDATE	DATE	7	N
8	SALARY	NUMBER	22	N
9	DISTANCEFROMHOSPITAL	FLOAT	22	N
10	DEPARTMENTID	NUMBER	22	N

```sql
CREATE TABLE Patients (
    PatientID INT NOT NULL,
    FirstName VARCHAR2(15) NOT NULL,
    LastName VARCHAR2(15) NOT NULL,
    DateOfBirth DATE NOT NULL,
    Gender CHAR(1) NOT NULL,
    Phone VARCHAR2(13) NOT NULL,
    AdmissionDate DATE NOT NULL,
    ReleaseDate DATE,
    Address VARCHAR2(15) NOT NULL,
    DepartmentID INT NOT NULL,
    PRIMARY KEY (PatientID),
    FOREIGN KEY (DepartmentID) REFERENCES Departments(DepartmentID)
);
```

	COLUMN_NAME	DATA_TYPE	DATA_LENGTH	NULLABLE
1	PATIENTID	NUMBER	22	N
2	FIRSTNAME	VARCHAR2	15	N
3	LASTNAME	VARCHAR2	15	N
4	DATEOFBIRTH	DATE	7	N
5	GENDER	CHAR	1	N
6	PHONE	VARCHAR2	13	N
7	ADMISSIONDATE	DATE	7	N
8	RELEASEDATE	DATE	7	Y
9	ADDRESS	VARCHAR2	15	N
10	DEPARTMENTID	NUMBER	22	N

```sql
CREATE TABLE PatientDoctor
(
  PatientID INT NOT NULL,
  DoctorID INT NOT NULL,
  PRIMARY KEY (PatientID, DoctorID),
  FOREIGN KEY (PatientID) REFERENCES Patients(PatientID),
  FOREIGN KEY (DoctorID) REFERENCES Doctors(DoctorID)
);
```

	COLUMN_NAME	DATA_TYPE	DATA_LENGTH	NULLABLE
1	PATIENTID	NUMBER	22	N
2	DOCTORID	NUMBER	22	N

הכנסת נתונים

1. סקריפט פייתון

```python
import random
import csv
import os
from datetime import datetime, timedelta

# Define the output path
output_path = r"C:\Users\noamm\Desktop\1 שלב\מיני פרוייקט בבסיסי נתונים\מיני פרוייקט ב\סמסטר ג\שנה ג\רימודים\pythonDataGenerated"


# Function to generate a phone number
2 usages
def generate_phone():
    return '05' + ''.join([str(random.randint(a: 0, b: 9)) for _ in range(8)])


# Function to generate random names
3 usages
def generate_name():
    first_names = ['John', 'Jane', 'Alice', 'David', 'Michael', 'Sara', 'James', 'Emily', 'Robert', 'Linda']
    last_names = ['Smith', 'Johnson', 'Williams', 'Jones', 'Brown', 'Davis', 'Miller', 'Wilson', 'Moore', 'Taylor']
    return random.choice(first_names), random.choice(last_names)


# Function to generate a random date between two dates
4 usages
def generate_date(start_date, end_date):
    delta = end_date - start_date
    random_days = random.randint(a: 0, delta.days)
    return start_date + timedelta(days=random_days)


```

```python
# Generate data for Departments table
with open(os.path.join(output_path, 'departments_data.txt'), 'w', newline='') as file:
    writer = csv.writer(file)
    writer.writerow(['DepartmentID', 'DepartmentName', 'BuildingName', 'Floor', 'Phone', 'TotalBeds', 'OccupiedBeds',
                     'HeadOfDepartment'])

    department_names = ['Gen Medicine', 'Pediatrics', 'Emerg Med', 'Surgery', 'Orthopedics', 'Obstetrics', 'Psychiatry',
                        'Neurology', 'Cardiology', 'Oncology', 'Dermatology', 'Urology', 'Gastroentero',
                        'Ophthalmology', 'Pulmonology', 'Rheumatology', 'Nephrology', 'Endocrinol', 'Infect Dis',
                        'Geriatrics']
    building_names = ['Building A', 'Building B', 'Building C', 'Building D', 'Building E']

    for department_id in range(1, 401):  # Generate 400 sample records
        department_name = random.choice(department_names)
        building_name = random.choice(building_names)
        floor = random.randint(a: 1, b: 10)
        phone = '02' + ''.join([str(random.randint(a: 0, b: 9)) for _ in range(7)])
        total_beds = random.randint(a: 20, b: 100)
        occupied_beds = random.randint(a: 0, total_beds)
        head_of_department = random.choice(generate_name()[1])  # Last name only

        writer.writerow([department_id, department_name, building_name, floor, phone, total_beds, occupied_beds,
                         head_of_department])

# Generate data for Doctors table
with open(os.path.join(output_path, 'doctors_data.txt'), 'w', newline='') as file:
    writer = csv.writer(file)
    writer.writerow(['DoctorID', 'FirstName', 'LastName', 'Specialty', 'Phone', 'DateOfBirth', 'HireDate', 'Salary',
                     'DistanceFromHospital', 'DepartmentID'])

```

```python
    specialties = ['Cardiology', 'Neurology', 'Oncology', 'Pediatrics', 'Surgery', 'Orthopedics']
    for doctor_id in range(1, 401):  # Generate 400 sample records
        first_name, last_name = generate_name()
        specialty = random.choice(specialties)
        phone = generate_phone()
        date_of_birth = generate_date(datetime( year: 1960,  month: 1,  day: 1), datetime( year: 2000,  month: 12,  day: 31))
        hire_date = generate_date(date_of_birth + timedelta(days=6570),
                                  datetime( year: 2023,  month: 1,  day: 1))  # Hire date is after 18 years old
        salary = random.randint( a: 50000,  b: 200000)
        distance_from_hospital = round(random.uniform( a: 1,  b: 100), 2)  # Max value of 100
        department_id = random.randint( a: 1,  b: 20)  # Assuming there are 20 departments

        writer.writerow([doctor_id, first_name, last_name, specialty, phone, date_of_birth.strftime('%Y-%m-%d'),
                         hire_date.strftime('%Y-%m-%d'), salary, distance_from_hospital, department_id])

# Generate data for Patients table
with open(os.path.join(output_path, 'patients_data.txt'), 'w', newline='') as file:
    writer = csv.writer(file)
    writer.writerow(
        ['PatientID', 'FirstName', 'LastName', 'DateOfBirth', 'Gender', 'Phone', 'AdmissionDate', 'ReleaseDate',
         'Address', 'DepartmentID'])

```

```python
    genders = ['M', 'F']
    for patient_id in range(1, 401):  # Generate 400 sample records
        first_name, last_name = generate_name()
        date_of_birth = generate_date(datetime( year: 1940,  month: 1,  day: 1), datetime( year: 2020,  month: 12,  day: 31))
        gender = random.choice(genders)
        phone = generate_phone()
        admission_date = generate_date(datetime( year: 2020,  month: 1,  day: 1), datetime( year: 2024,  month: 1,  day: 1))
        release_date = admission_date + timedelta(days=random.randint( a: 1,  b: 30))
        address = f"{random.randint( a: 1,  b: 999)} Elm St"
        department_id = random.randint( a: 1,  b: 20)  # Assuming there are 20 departments

        writer.writerow([patient_id, first_name, last_name, date_of_birth.strftime('%Y-%m-%d'), gender, phone,
                         admission_date.strftime('%Y-%m-%d'),
                         release_date.strftime('%Y-%m-%d') if random.random() > 0.5 else '', address, department_id])

# Generate data for PatientDoctor table
with open(os.path.join(output_path, 'patientdoctor_data.txt'), 'w', newline='') as file:
    writer = csv.writer(file)
    writer.writerow(['PatientID', 'DoctorID'])

    for patient_id in range(1, 401):  # Assuming each patient can have multiple doctors
        doctor_ids = random.sample(range(1, 401), random.randint( a: 1,  b: 3))  # Each patient can have 1 to 3 doctors
        for doctor_id in doctor_ids:
            writer.writerow([patient_id, doctor_id])

print(f"Data generation complete. Files created at: {output_path}")

```

```
DepartmentID,DepartmentName,BuildingName,Floor,Phone,TotalBeds,OccupiedBeds,HeadOfDepartment
1001,Rheumatology,Building E,6,025733112,94,62,Alice Williams
1002,Emergency Med,Building D,1,026925363,72,20,Sara Miller
1003,Pediatrics,Building C,1,021499191,22,10,Linda Taylor
1004,Neurology,Building B,10,028776100,45,8,Emily Wilson
1005,Infectious,Building E,7,027938696,40,31,Sara Davis
1006,Geriatrics,Building C,8,023993537,49,35,John Miller
1007,Rheumatology,Building E,7,023030501,76,20,Robert Smith
1008,Gen Medicine,Building B,4,021368281,57,27,Linda Smith
1009,Ophthalmology,Building A,3,025676659,88,41,John Williams
```

Configuration

General

Fieldcount: 8

Quote character: "

Comment line:

Import lines: 1 ..

- [x] End at line-end
- [x] Name in header
- [x] Skip empty lines

Field1 (+0 .. ",") DepartmentID
Field2 (+0 .. ",") DepartmentNam
Field3 (+0 .. ",") BuildingName
Field4 (+0 .. ",") Floor
Field5 (+0 .. ",") Phone
Field6 (+0 .. ",") TotalBeds
Field7 (+0 .. ",") OccupiedBeds
Field8 (+0 .. ",") HeadOfDepartm

Field Start
- (•) Relative position: 0
- ( ) Absolute position
- ( ) Character

Field End
- ( ) Length
- (•) Character: ,

Filter

Result Preview

DepartmentID	DepartmentName	BuildingName	Floor	Phone	TotalBeds	OccupiedBeds	HeadOfDepartment
1001	Rheumatology	Building E	6	025733112	94	62	Alice Williams
1002	Emergency Med	Building D	1	026925363	72	20	Sara Miller

Select departments | Select doctors | Select patients | Select patientdoctor

	DEPARTMENTID	DEPARTMENTNAME	BUILDINGNAME	FLOOR	PHONE	TOTALBEDS	OCCUPIEDBEDS	HEADOFDEPARTMENT
1	1102	Ophthalmology	Building C	9	020318304	44	44	Wilson
2	1103	Orthopedics	Building D	4	022726618	78	59	Smith
3	1104	Rheumatology	Building B	9	022497752	20	9	Moore
4	1001	Ophthalmology	Building D	8	021715150	26	11	Brown
5	1002	Cardiology	Building D	2	021465138	61	5	Miller
6	1003	Endocrinology	Building C	8	028070012	40	34	Davis
7	1004	Nephrology	Building A	10	022810815	44	18	Jones
8	1005	Nephrology	Building A	4	025196326	22	8	Jones
9	1006	Dermatology	Building A	7	028525729	20	5	Williams
10	1007	Pediatrics	Building B	9	022159936	42	25	Miller
11	1008	Emergency Med	Building C	1	028222432	24	5	Miller
12	1009	Surgery	Building B	7	026718267	32	12	Wilson
13	1010	Neurology	Building A	2	020289577	47	14	Johnson
14	1011	Emergency Med	Building B	8	025070236	32	15	Williams
15	1012	Oncology	Building D	2	024017155	95	81	Brown
16	1013	Emergency Med	Building B	4	020811216	25	13	Davis
17	1014	Nephrology	Building D	3	021867512	89	10	Davis
18	1015	Urology	Building A	8	024041688	57	36	Moore

1 of 400 | nmandelb AS SYSDBA | [14:26:09] 400 rows selected in 0.431 seconds

2. פקודות INSERT

```oracle
-- Inserting data into the Departments table
INSERT INTO Departments (DepartmentID, DepartmentName, BuildingName, Floor, Phone, TotalBeds, OccupiedBeds, HeadOfDepartment) VALUES
(1, 'Cardiology', 'Building A', 1, '0256739812', 50, 30, 'John Doe');
INSERT INTO Departments (DepartmentID, DepartmentName, BuildingName,Floor , Phone, TotalBeds, OccupiedBeds, HeadOfDepartment) VALUES
(2, 'Neurology', 'Building B', 2, '0256739813', 40, 20, 'Jane Smith');
INSERT INTO Departments (DepartmentID, DepartmentName, BuildingName, Floor, Phone, TotalBeds, OccupiedBeds, HeadOfDepartment) VALUES
(3, 'Oncology', 'Building C', 3, '0256739814', 60, 40, 'Jim Beam');

-- Inserting data into the Doctors table
INSERT INTO Doctors (DoctorID, FirstName, LastName, Specialty, Phone, DateOfBirth, HireDate, Salary, DistanceFromHospital, DepartmentID) VALUES
(1, 'Alice', 'Johnson', 'Cardiology', '0587677615', TO_DATE('1975-05-20', 'YYYY-MM-DD'), TO_DATE('2000-06-15', 'YYYY-MM-DD'), 120000, 10, 1);
INSERT INTO Doctors (DoctorID, FirstName, LastName, Specialty, Phone, DateOfBirth, HireDate, Salary, DistanceFromHospital, DepartmentID) VALUES
(2, 'Bob', 'Williams', 'Neurology', '0587677616', TO_DATE('1980-11-10', 'YYYY-MM-DD'), TO_DATE('2005-09-01', 'YYYY-MM-DD'), 110000, 20, 2);
INSERT INTO Doctors (DoctorID, FirstName, LastName, Specialty, Phone, DateOfBirth, HireDate, Salary, DistanceFromHospital, DepartmentID) VALUES
(3, 'Charlie', 'Brown', 'Oncology', '0587677617', TO_DATE('1985-02-25', 'YYYY-MM-DD'), TO_DATE('2010-03-20', 'YYYY-MM-DD'), 130000, 15, 3);

-- Inserting data into the Patients table
INSERT INTO Patients (PatientID, FirstName, LastName, DateOfBirth, Gender, Phone, AdmissionDate, ReleaseDate, Address, DepartmentID) VALUES
(1, 'David', 'Smith', TO_DATE('1990-04-12', 'YYYY-MM-DD'), 'M', '0587677618', TO_DATE('2024-07-01', 'YYYY-MM-DD'), TO_DATE('2024-07-10', 'YYYY-MM-DD'), '123 Elm St', 1);
INSERT INTO Patients (PatientID, FirstName, LastName, DateOfBirth, Gender, Phone, AdmissionDate, ReleaseDate, Address, DepartmentID) VALUES
(2, 'Eva', 'Green', TO_DATE('1985-08-15', 'YYYY-MM-DD'), 'F', '0587677619', TO_DATE('2024-06-20', 'YYYY-MM-DD'), TO_DATE('2024-07-05', 'YYYY-MM-DD'), '456 Oak St', 2);
INSERT INTO Patients (PatientID, FirstName, LastName, DateOfBirth, Gender, Phone, AdmissionDate, ReleaseDate, Address, DepartmentID) VALUES
(3, 'Frank', 'White', TO_DATE('1978-12-22', 'YYYY-MM-DD'), 'M', '0587677620', TO_DATE('2024-05-10', 'YYYY-MM-DD'), TO_DATE('2024-05-20', 'YYYY-MM-DD'), '789 Pine St', 3);

-- Inserting data into the Patients table
INSERT INTO Patients (PatientID, FirstName, LastName, DateOfBirth, Gender, Phone, AdmissionDate, ReleaseDate, Address, DepartmentID) VALUES
(1, 'David', 'Smith', TO_DATE('1990-04-12', 'YYYY-MM-DD'), 'M', '0587677618', TO_DATE('2024-07-01', 'YYYY-MM-DD'), TO_DATE('2024-07-10', 'YYYY-MM-DD'), '123 Elm St', 1);
INSERT INTO Patients (PatientID, FirstName, LastName, DateOfBirth, Gender, Phone, AdmissionDate, ReleaseDate, Address, DepartmentID) VALUES
(2, 'Eva', 'Green', TO_DATE('1985-08-15', 'YYYY-MM-DD'), 'F', '0587677619', TO_DATE('2024-06-20', 'YYYY-MM-DD'), TO_DATE('2024-07-05', 'YYYY-MM-DD'), '456 Oak St', 2);
INSERT INTO Patients (PatientID, FirstName, LastName, DateOfBirth, Gender, Phone, AdmissionDate, ReleaseDate, Address, DepartmentID) VALUES
(3, 'Frank', 'White', TO_DATE('1978-12-22', 'YYYY-MM-DD'), 'M', '0587677620', TO_DATE('2024-05-10', 'YYYY-MM-DD'), TO_DATE('2024-05-20', 'YYYY-MM-DD'), '789 Pine St', 3);

-- Inserting data into the PatientDoctor table
INSERT INTO PatientDoctor (PatientID, DoctorID) VALUES
(1, 1);
INSERT INTO PatientDoctor (PatientID, DoctorID) VALUES
(2, 2);
INSERT INTO PatientDoctor (PatientID, DoctorID) VALUES
(3, 3);
COMMIT;
```

Select departments | Select doctors | Select patients | Select patientdoctor

	DEPARTMENTID	DEPARTMENTNAME	BUILDINGNAME	FLOOR	PHONE	TOTALBEDS	OCCUPIEDBEDS	HEADOFDEPARTMENT
1	1	Cardiology	Building A	1	0256739812	50	30	John Doe
2	2	Neurology	Building B	2	0256739813	40	20	Jane Smith
3	3	Oncology	Building C	3	0256739814	60	40	Jim Beam

Select departments | Select doctors | Select patients | Select patientdoctor

	DOCTORID	FIRSTNAME	LASTNAME	SPECIALTY	PHONE	DATEOFBIRTH	HIREDATE	SALARY	DISTANCEFROMHOSPITAL	DEPARTMENTID
1	1	Alice	Johnson	Cardiology	0587677615	20/05/1975	15/06/2000	120000	10	1
2	2	Bob	Williams	Neurology	0587677616	10/11/1980	01/09/2005	110000	20	2
3	3	Charlie	Brown	Oncology	0587677617	25/02/1985	20/03/2010	130000	15	3

Select departments | Select doctors | Select patients | Select patientdoctor

	PATIENTID	FIRSTNAME	LASTNAME	DATEOFBIRTH	GENDER	PHONE	ADMISSIONDATE	RELEASEDATE	ADDRESS	DEPARTMENTID
1	1	David	Smith	12/04/1990	M	0587677618	01/07/2024	10/07/2024	123 Elm St	1
2	2	Eva	Green	15/08/1985	F	0587677619	20/06/2024	05/07/2024	456 Oak St	2
3	3	Frank	White	22/12/1978	M	0587677620	10/05/2024	20/05/2024	789 Pine St	3

Select departments | Select doctors | Select patients | Select patientdoctor

	PATIENTID	DOCTORID
1	1	1
2	2	2
3	3	3

בשביל לדמות את הנתונים למציאות:

1. נשתמש ב20 המחלקות הראשונות בלבד ונבצע DELETE על המחלקות שלא נשתמש בהן:

```
DELETE FROM Departments
WHERE DepartmentID > 20;
```

2. נמחק את כל הרופאים שלא טיפלו כלומר שאינם בטבלת PatientDoctor:

```
DELETE FROM Doctors d
WHERE NOT EXISTS (
    SELECT 1
    FROM PatientDoctor pd
    WHERE pd.DoctorID = d.DoctorID
);
```

שלב ב'

שאילתות SELECT

1. בית החולים רוצה לדעת אילו מחלקות עשויות להזדקק לתגבור של רופאים בשעת חירום, על ידי ניתוח היחס בין מספר המטופלים למספר הרופאים הגרים בטווח של 20 ק"מ, והבנת עומס העבודה בכל מחלקה.

```
SELECT
    dp.DepartmentID,
    dp.DepartmentName,
    dp.TotalBeds,
    dp.OccupiedBeds,
    COUNT(DISTINCT d.DoctorID) AS NumberOfNearbyDoctors,
    COUNT(DISTINCT p.PatientID) AS NumberOfPatients,
    (COUNT(DISTINCT p.PatientID) / NULLIF(COUNT(DISTINCT d.DoctorID), 0)) AS PatientsPerDoctor,
    (dp.OccupiedBeds / NULLIF(dp.TotalBeds, 0)) * 100 AS OccupancyRate,
    dp.HeadOfDepartment,
    dp.Phone
FROM
    Departments dp
    LEFT JOIN Doctors d ON dp.DepartmentID = d.DepartmentID AND d.DistanceFromHospital <= 20
    LEFT JOIN Patients p ON dp.DepartmentID = p.DepartmentID AND p.ReleaseDate IS NULL
WHERE
    dp.TotalBeds IS NOT NULL AND dp.OccupiedBeds IS NOT NULL
GROUP BY
    dp.DepartmentID,
    dp.DepartmentName,
    dp.TotalBeds,
    dp.OccupiedBeds,
    dp.HeadOfDepartment,
    dp.Phone
HAVING
    (COUNT(DISTINCT d.DoctorID) = 0)
    OR ((COUNT(DISTINCT p.PatientID) / NULLIF(COUNT(DISTINCT d.DoctorID), 0)) > 5)
ORDER BY
    OccupancyRate DESC,
    PatientsPerDoctor DESC;
```

	DEPARTMENTID	DEPARTMENTNAME	TOTALBEDS	OCCUPIEDBEDS	NUMBEROFNEARBYDOCTORS	NUMBEROFPATIENTS	PATIENTSPERDOCTOR	OCCUPANCYRATE	HEADOFDEPARTMENT	PHONE
1	10	Orthopedics	92	51	8	51	6.375	55.4347826086956	David Wilson	022516288
2	6	Gastroentero	89	44	5	44	8.8	49.438202247191	Sara Williams	029191634
3	14	Rheumatology	99	46	3	46	15.3333333333333	46.4646464646465	Jane Taylor	021087961
4	7	Nephrology	96	42	8	42	5.25	43.75	Alice Moore	028011507
5	12	Oncology	65	26	2	26	13	40	Jane Taylor	028265263

2. בית החולים מעוניין לחגוג ימי הולדת של רופאים בחודש הקרוב, להכיר את המטופלים שהם מטפלים בהם, ולארגן את המידע לפי מחלקות לצורך תכנון אירועים במחלקות השונות.

```sql
SELECT
    d.DoctorID,
    d.FirstName || ' ' || d.LastName AS DoctorName,
    dp.DepartmentID,
    dp.DepartmentName,
    EXTRACT(DAY FROM d.DateOfBirth) AS BirthDay,
    EXTRACT(MONTH FROM d.DateOfBirth) AS BirthMonth,
    COUNT(DISTINCT p.PatientID) AS NumberOfCurrentPatients
FROM
    Doctors d
    JOIN Departments dp ON d.DepartmentID = dp.DepartmentID
    LEFT JOIN PatientDoctor pd ON d.DoctorID = pd.DoctorID
    LEFT JOIN Patients p ON pd.PatientID = p.PatientID AND p.ReleaseDate IS NULL
WHERE
    EXTRACT(MONTH FROM d.DateOfBirth) = EXTRACT(MONTH FROM SYSDATE)
    OR EXTRACT(MONTH FROM d.DateOfBirth) = EXTRACT(MONTH FROM ADD_MONTHS(SYSDATE, 1))
GROUP BY
    d.DoctorID,
    d.FirstName,
    d.LastName,
    dp.DepartmentID,
    dp.DepartmentName,
    d.DateOfBirth
ORDER BY
    dp.DepartmentID,
    BirthMonth,
    BirthDay;
```

	DOCTORID	DOCTORNAME	DEPARTMENTID	DEPARTMENTNAME	BIRTHDAY	BIRTHMONTH	NUMBEROFCURRENTPATIENTS
1	309815512	Michael Moore	1	Pediatrics	2	9	0
2	305317262	James Moore	1	Pediatrics	18	9	6
3	209531628	Linda Moore	1	Pediatrics	25	9	1
4	208771808	Sara Williams	1	Pediatrics	4	10	3
5	307844344	Jane Wilson	1	Pediatrics	11	10	1

3. בית החולים מעוניין בניתוח מחלקות לפי השכר והניסיון של הרופאים שעובדים בהן - אלו מהמחלקות משלמות הכי הרבה ביחס לשנות הניסיון של רופאיה, תוך בחינת חמש המחלקות המובילות בקטגוריה זו.

```sql
-- Query 3: Identify departments with high patient load where most doctors have less than
-- two years of experience to assess training or staffing needs.
SELECT
    dp.DepartmentID,
    dp.DepartmentName,
    ROUND(AVG(d.Salary), 2) AS AverageSalary,
    ROUND(AVG(MONTHS_BETWEEN(SYSDATE, d.HireDate) / 12), 2) AS AverageExperience,
    ROUND((AVG(d.Salary) / NULLIF(AVG(MONTHS_BETWEEN(SYSDATE, d.HireDate) / 12), 0)), 2) AS SalaryPerYearExperience,
    COUNT(d.DoctorID) AS NumberOfDoctors
FROM
    Departments dp
    JOIN Doctors d ON dp.DepartmentID = d.DepartmentID
GROUP BY
    dp.DepartmentID,
    dp.DepartmentName
HAVING
    COUNT(d.DoctorID) > 0
ORDER BY
    SalaryPerYearExperience DESC
FETCH FIRST 5 ROWS ONLY;
```

Select departments | Select dateofbirth | Select departments | Select departments

	DEPARTMENTID	DEPARTMENTNAME	AVERAGESALARY	AVERAGEEXPERIENCE	SALARYPERYEAREXPERIENCE	NUMBEROFDOCTORS
1	5	Cardiology	135664.06	11.21	12099.66	18
2	9	Pediatrics	121519.82	10.51	11562.37	17
3	14	Rheumatology	130856.33	11.86	11034.1	15
4	20	Geriatrics	111961.57	10.49	10675.76	14
5	17	Surgery	108823.59	10.46	10407.9	17

4. בית החולים מעוניין לדעת כמה רופאים מעל גיל 60 יש בכל מחלקה, להבין את העומס עליהם מבחינת מספר המטופלים, ולבחון האם יש צורך בהכשרת רופאים צעירים כדי להבטיח רציפות טיפולית לקראת פרישתם.

```sql
SELECT
    dp.DepartmentID,
    dp.DepartmentName,
    COUNT(d.DoctorID) AS TotalDoctors,
    COUNT(CASE WHEN FLOOR(MONTHS_BETWEEN(SYSDATE, d.DateOfBirth) / 12) >= 60 THEN d.DoctorID END) AS DoctorsOver60,
    ROUND(AVG(FLOOR(MONTHS_BETWEEN(SYSDATE, d.DateOfBirth) / 12)), 2) AS AverageAge,
    ROUND(AVG(CASE WHEN FLOOR(MONTHS_BETWEEN(SYSDATE, d.DateOfBirth) / 12) >= 60 THEN d.Salary END), 2) AS AverageSalaryOver60,
    ROUND(AVG(CASE WHEN d_over60.DoctorID IS NOT NULL THEN d_over60.NumPatients END), 2) AS AveragePatientsPerDoctorOver60
FROM
    Departments dp
    JOIN Doctors d ON dp.DepartmentID = d.DepartmentID
    LEFT JOIN (
        SELECT
            d.DoctorID,
            COUNT(DISTINCT pd.PatientID) AS NumPatients
        FROM
            Doctors d
            LEFT JOIN PatientDoctor pd ON d.DoctorID = pd.DoctorID
        WHERE
            FLOOR(MONTHS_BETWEEN(SYSDATE, d.DateOfBirth) / 12) >= 60
        GROUP BY
            d.DoctorID
    ) d_over60 ON d.DoctorID = d_over60.DoctorID
GROUP BY
    dp.DepartmentID,
    dp.DepartmentName
HAVING
    COUNT(CASE WHEN FLOOR(MONTHS_BETWEEN(SYSDATE, d.DateOfBirth) / 12) >= 60 THEN d.DoctorID END) > 0
ORDER BY
    DoctorsOver60 DESC;
```

	DEPARTMENTID	DEPARTMENTNAME	TOTALDOCTORS	DOCTORSOVER60	AVERAGEAGE	AVERAGESALARYOVER60	AVERAGEPATIENTSPERDOCTOROVER60
1	3	Gastroentero	21	7	47.62	126638.43	7.43
2	7	Nephrology	22	4	44.05	139911.5	6.75
3	11	Gastroentero	26	4	45.65	168686.25	4.75
4	8	Dermatology	18	4	44.22	87559.5	3.5
5	4	Geriatrics	27	3	43.59	117333	2.33

שאילתות DELETE

1. הנהלת בית החולים רוצה למחוק את הרשומות של כל מטופל ששוחרר לפני יותר מ3 שנים ולא אושפז פעם נוספת לאחר מכן, בתנאי שכל הרופאים שטיפלו בו עובדים בבית החולים יותר מ5 שנים.

לפני המחיקה

```
SELECT p.PatientID, p.FirstName, p.LastName, p.ReleaseDate
FROM Patients p
WHERE p.ReleaseDate < ADD_MONTHS(SYSDATE, -36)  -- שנים 3-שוחררו לפני יותר מ
AND NOT EXISTS (  -- בדיקה שהמטופל לא טופל שוב לאחר השחרור
    SELECT 1
    FROM PatientDoctor pd2
    JOIN Patients p2 ON pd2.PatientID = p2.PatientID
    WHERE pd2.PatientID = p.PatientID
    AND p2.AdmissionDate > p.ReleaseDate
)
AND NOT EXISTS (  -- בדיקה אם יש רופא שלא עומד בתנאי הוותק של 5 שנים
    SELECT 1
    FROM PatientDoctor pd
    JOIN Doctors d ON pd.DoctorID = d.DoctorID
    WHERE pd.PatientID = p.PatientID
    AND d.HireDate >= ADD_MONTHS(SYSDATE, -60)  -- שנים 5-רופא שעובד פחות מ
);
```

Select patients | Delete patients | Select patients | Rollback

	PATIENTID	FIRSTNAME	LASTNAME	RELEASEDATE
1	307193006	Michael	Johnson	16/01/2021
2	205595060	Linda	Miller	18/04/2021
3	207482257	Emily	Brown	11/09/2020
4	309606147	David	Jones	10/04/2021

מחיקה

```sql
DELETE FROM Patients p
WHERE p.PatientID IN (
    SELECT p.PatientID
    FROM Patients p
    WHERE p.ReleaseDate < ADD_MONTHS(SYSDATE, -36)  -- שנים 3-שוחררו לפני יותר מ
    AND NOT EXISTS (  -- בדיקה שהמטופל לא טופל שוב לאחר השחרור
        SELECT 1
        FROM PatientDoctor pd2
        JOIN Patients p2 ON pd2.PatientID = p2.PatientID
        WHERE pd2.PatientID = p.PatientID
        AND p2.AdmissionDate > p.ReleaseDate
    )
    AND NOT EXISTS (  -- בדיקה אם יש רופא שלא עומד בתנאי הוותק של 5 שנים
        SELECT 1
        FROM PatientDoctor pd
        JOIN Doctors d ON pd.DoctorID = d.DoctorID
        WHERE pd.PatientID = p.PatientID
        AND d.HireDate >= ADD_MONTHS(SYSDATE, -60)  -- שנים 5-רופא שעובד פחות מ
    )
);
```

Select patients | Delete patients | Select patients | Rollback

אחרי המחיקה

```sql
SELECT p.PatientID, p.FirstName, p.LastName, p.ReleaseDate
FROM Patients p
WHERE p.ReleaseDate < ADD_MONTHS(SYSDATE, -36)  -- שנים 3-שוחררו לפני יותר מ
AND NOT EXISTS (  -- בדיקה שהמטופל לא טופל שוב לאחר השחרור
    SELECT 1
    FROM PatientDoctor pd2
    JOIN Patients p2 ON pd2.PatientID = p2.PatientID
    WHERE pd2.PatientID = p.PatientID
    AND p2.AdmissionDate > p.ReleaseDate
)
AND NOT EXISTS (  -- בדיקה אם יש רופא שלא עומד בתנאי הוותק של 5 שנים
    SELECT 1
    FROM PatientDoctor pd
    JOIN Doctors d ON pd.DoctorID = d.DoctorID
    WHERE pd.PatientID = p.PatientID
    AND d.HireDate >= ADD_MONTHS(SYSDATE, -60)  -- שנים 5-רופא שעובד פחות מ
);
```

Select patients | Delete patients | Select patients | Rollback

PATIENTID	FIRSTNAME	LASTNAME	RELEASEDATE

2. כחלק מתהליך התייעלות, בית החולים מעוניין למחוק רופאים ממחלקות פחות פעילות שאין להם מטופלים.

לפני המחיקה

```
-- Query 2:

SELECT d.DoctorID, d.FirstName || ' ' || d.LastName AS DoctorName, d.DepartmentID, dep.DepartmentName, dep.OccupiedBeds
FROM Doctors d
LEFT JOIN PatientDoctor pd ON d.DoctorID = pd.DoctorID
JOIN Departments dep ON d.DepartmentID = dep.DepartmentID
WHERE pd.PatientID IS NULL
  AND dep.OccupiedBeds < (
      SELECT AVG(OccupiedBeds) FROM Departments
  )
ORDER BY d.DepartmentID;
```

Select patients | Delete patients | Select patients | Select doctors | Delete doctors | Select doctors | Rollback

	DOCTORID	DOCTORNAME	DEPARTMENTID	DEPARTMENTNAME	OCCUPIEDBEDS
1	306029277	Emily Williams	4	Geriatrics	16
2	304423621	James Williams	4	Geriatrics	16
3	302503504	Emily Brown	4	Geriatrics	16
4	307245824	James Davis	5	Cardiology	6
5	203505934	John Miller	9	Pediatrics	11

מחיקה

```
DELETE FROM Doctors
WHERE DoctorID IN (
    SELECT d.DoctorID
    FROM Doctors d
    LEFT JOIN PatientDoctor pd ON d.DoctorID = pd.DoctorID
    JOIN Departments dep ON d.DepartmentID = dep.DepartmentID
    WHERE pd.PatientID IS NULL
      AND dep.OccupiedBeds < (
          SELECT AVG(OccupiedBeds) FROM Departments
      )
);
```

Select patients | Delete patients | Select patients | Select doctors | Delete doctors | Select doctors | Rollback

אחרי המחיקה

```
SELECT d.DoctorID, d.FirstName || ' ' || d.LastName AS DoctorName, d.DepartmentID, dep.DepartmentName, dep.OccupiedBeds
FROM Doctors d
LEFT JOIN PatientDoctor pd ON d.DoctorID = pd.DoctorID
JOIN Departments dep ON d.DepartmentID = dep.DepartmentID
WHERE pd.PatientID IS NULL
  AND dep.OccupiedBeds < (
      SELECT AVG(OccupiedBeds) FROM Departments
  )
ORDER BY d.DepartmentID;
```

Select patients | Delete patients | Select patients | Select doctors | Delete doctors | Select doctors | Rollback

DOCTORID	DOCTORNAME	DEPARTMENTID	DEPARTMENTNAME	OCCUPIEDBEDS

שאילתות UPDATE

1. הנהלת בית החולים רוצה להוסיף 10% למשכורות של עשרת הרופאים שטיפלו בהכי הרבה מטופלים ב3 השנים האחרונות.

לפני העדכון:

	DOCTORID	DOCTORNAME	SALARY	NUMPATIENTS	DEPARTMENTID	DEPARTMENTNAME
1	307029243	Michael Johnson	65501	13	14	Rheumatology
2	301084401	Robert Davis	53425	12	10	Orthopedics
3	302675506	Robert Jones	100659	12	20	Geriatrics
4	303862123	Jane Williams	145126	12	12	Oncology
5	200324112	Robert Brown	134540	11	14	Rheumatology
6	201608661	David Smith	115355	11	3	Gastroentero
7	208885586	Sara Miller	142472	11	14	Rheumatology
8	302789386	James Brown	100464	11	10	Orthopedics
9	309347888	Alice Jones	83354	11	20	Geriatrics
10	202417459	Alice Brown	99780	10	10	Orthopedics

עדכון:

```
UPDATE Doctors
SET Salary = Salary * 1.1
WHERE DoctorID IN (
    SELECT DoctorID
    FROM (
        SELECT d.DoctorID, COUNT(pd.PatientID) AS NumPatients
        FROM Doctors d
        JOIN PatientDoctor pd ON d.DoctorID = pd.DoctorID
        JOIN Patients p ON pd.PatientID = p.PatientID
        JOIN Departments dep ON d.DepartmentID = dep.DepartmentID
        WHERE EXTRACT(YEAR FROM p.AdmissionDate) BETWEEN EXTRACT(YEAR FROM SYSDATE) - 3 AND EXTRACT(YEAR FROM SYSDATE)
        GROUP BY d.DoctorID, d.DepartmentID, dep.DepartmentName
        ORDER BY NumPatients DESC, d.DoctorID ASC
        FETCH FIRST 10 ROWS ONLY
    )
);
```

Select doctors | Update doctors | Select doctors | Rollback

לאחר העדכון:

	DOCTORID	DOCTORNAME	SALARY	NUMPATIENTS	DEPARTMENTID	DEPARTMENTNAME
1	307029243	Michael Johnson	72051	13	14	Rheumatology
2	301084401	Robert Davis	58768	12	10	Orthopedics
3	302675506	Robert Jones	110725	12	20	Geriatrics
4	303862123	Jane Williams	159639	12	12	Oncology
5	200324112	Robert Brown	147994	11	14	Rheumatology
6	201608661	David Smith	126891	11	3	Gastroentero
7	208885586	Sara Miller	156719	11	14	Rheumatology
8	302789386	James Brown	110510	11	10	Orthopedics
9	309347888	Alice Jones	91689	11	20	Geriatrics
10	202417459	Alice Brown	109758	10	10	Orthopedics

2. בית החולים רוצה לוודא שהשכר של הרופאים תואם את הוותק שלהם. רופאים שעובדים במחלקות שבהן יחס השכר הממוצע לוותק נמוך מהממוצע הכללי, יקבלו העלאה של 15% בשכרם.

לפני העדכון:

	DEPARTMENTID	DEPARTMENTNAME	AVERAGESALARY	AVERAGEEXPERIENCE	SALARYPERYEAREXPERIENCE
1	1	Pediatrics	129182.93	12.51	10328.09
2	15	Neurology	126526.62	20.51	6168.21
3	7	Nephrology	136205.55	14.33	9502.35
4	13	Nephrology	122278.19	15.26	8013.74
5	8	Dermatology	112722.17	13.89	8115
6	17	Surgery	108823.59	10.46	10402.77
7	14	Rheumatology	133139.73	11.86	11221.76
8	18	Obstetrics	93458.06	13.59	6875.47
9	4	Geriatrics	118292.07	16.56	7144.39
10	2	Rheumatology	122815.33	13.35	9202.01
11	3	Gastroentero	117576.86	17.75	6623.45
12	9	Pediatrics	121519.82	10.52	11556.79
13	10	Orthopedics	124773.05	17.54	7114.94
14	19	Endocrinol	112046.1	13.44	8335.23
15	16	Geriatrics	127269.08	13.53	9404.99
16	11	Gastroentero	136580.54	14.42	9471.05
17	5	Cardiology	135664.06	11.22	12094.19
18	20	Geriatrics	113275.93	10.49	10795.89
19	6	Gastroentero	124463.33	14.22	8750.82
20	12	Oncology	127873.2	16.85	7588.43

עדכון:

```
UPDATE Doctors d
SET d.Salary = d.Salary * 1.15
WHERE d.DepartmentID IN (
    SELECT dp.DepartmentID
    FROM Departments dp
    JOIN Doctors d2 ON dp.DepartmentID = d2.DepartmentID
    GROUP BY dp.DepartmentID
    HAVING ROUND(AVG(d2.Salary) / NULLIF(AVG(MONTHS_BETWEEN(SYSDATE, d2.HireDate) / 12), 0), 2) < (
        SELECT ROUND(AVG(d3.Salary) / NULLIF(AVG(MONTHS_BETWEEN(SYSDATE, d3.HireDate) / 12), 0), 2)
        FROM Doctors d3
    )
);
```

Select doctors | Update doctors | Select doctors | Select departments | Update doctors | Select departments | Rollback

לאחר עדכון:

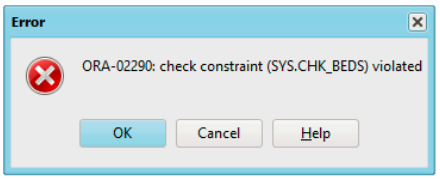
	DEPARTMENTID	DEPARTMENTNAME	AVERAGESALARY	AVERAGEEXPERIENCE	SALARYPERYEAREXPERIENCE
1	1	Pediatrics	129182.93	12.51	10328.09
2	15	Neurology	145505.69	20.51	7093.44
3	7	Nephrology	136205.55	14.33	9502.35
4	13	Nephrology	140619.94	15.26	9215.81
5	8	Dermatology	129630.5	13.89	9332.25
6	17	Surgery	108823.59	10.46	10402.77
7	14	Rheumatology	133139.73	11.86	11221.76
8	18	Obstetrics	107476.76	13.59	7906.79
9	4	Geriatrics	136035.89	16.56	8216.05
10	2	Rheumatology	122815.33	13.35	9202.01
11	3	Gastroentero	135213.43	17.75	7616.97
12	9	Pediatrics	121519.82	10.52	11556.79
13	10	Orthopedics	143488.95	17.54	8182.18
14	19	Endocrinol	128853.1	13.44	9585.52
15	16	Geriatrics	127269.08	13.53	9404.99
16	11	Gastroentero	136580.54	14.42	9471.05
17	5	Cardiology	135664.06	11.22	12094.19
18	20	Geriatrics	113275.93	10.49	10795.89
19	6	Gastroentero	124463.33	14.22	8750.82
20	12	Oncology	147054.13	16.85	8726.69

הוספת אילוצים

1. בדיקה שמספר המיטות המאויישות במחלקה קטן או שווה למספר המיטות המקסימלי במחלקה

```
ALTER TABLE Departments
ADD CONSTRAINT chk_beds CHECK (TotalBeds >= OccupiedBeds);
```

```
INSERT INTO Departments (DepartmentID, DepartmentName, BuildingName, Floor, Phone, TotalBeds, OccupiedBeds, HeadOfDepartment)
VALUES (21, 'Cardiology', 'A', 1, '02-1234567', 10, 15, 'Dr. Cohen');
```



2. נגדיר שבברירת מחדל משכורת רופא היא 5000

```
ALTER TABLE Doctors
MODIFY Salary DEFAULT 5000;
```

```
INSERT INTO Doctors (DoctorID, FirstName, LastName, Specialty, Phone, DateOfBirth, HireDate, DistanceFromHospital, DepartmentID)
VALUES (123, 'John', 'Doe', 'Cardiology', '555-1234', TO_DATE('1985-06-15', 'YYYY-MM-DD'), TO_DATE('2023-09-01', 'YYYY-MM-DD'), 25.0, 1);
```

```
SELECT *
FROM Doctors
WHERE DoctorID = 123;
```

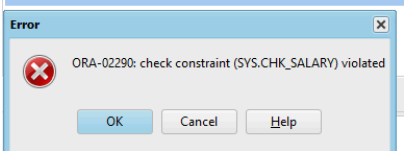
Insert doctors | Select doctors

	DOCTORID	FIRSTNAME	LASTNAME	SPECIALTY	PHONE	DATEOFBIRTH	HIREDATE	SALARY	DISTANCEFROMHOSPITAL	DEPARTMENTID
1	123	John	Doe	Cardiology	555-1234	15/06/1985	01/09/2023	5000	25	1

3. נגדיר בדיקה שערך משכורת רופא אינו שלילי

```
ALTER TABLE Doctors
ADD CONSTRAINT chk_salary CHECK (Salary > 0);
```

```
INSERT INTO Doctors (DoctorID, FirstName, LastName, Specialty, Phone, DateOfBirth, HireDate, Salary, DistanceFromHospital, DepartmentID)
VALUES (124, 'Jane', 'Doe', 'Neurology', '555-5678', TO_DATE('1990-04-22', 'YYYY-MM-DD'), TO_DATE('2023-09-01', 'YYYY-MM-DD'), -1000, 30.0, 2);
```



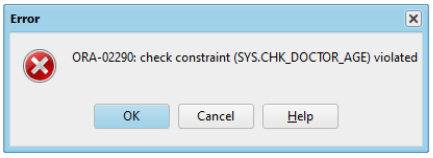
4. בדיקה שמרחק המגורים של רופא מבית החולים אינו עולה על 100 ק"מ

```
ALTER TABLE Doctors
ADD CONSTRAINT chk_doctor_distance CHECK (
    DistanceFromHospital <= 100
);
```

5. בדיקה שגילו של רופא אינו קטן מ18

```sql
ALTER TABLE Doctors
ADD CONSTRAINT chk_doctor_age CHECK (
    (HireDate - DateOfBirth) / 365 >= 18
);
```

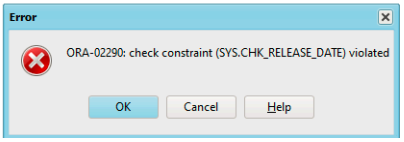
```sql
INSERT INTO Doctors (DoctorID, FirstName, LastName, Specialty, Phone, DateOfBirth, HireDate, Salary, DistanceFromHospital, DepartmentID)
VALUES (200, 'Young', 'Doctor', 'Surgery', '555-1234', TO_DATE('2000-01-01', 'YYYY-MM-DD'), TO_DATE('2017-09-01', 'YYYY-MM-DD'), 10000, 50, 1);
```



6. בדיקה שתאריך השחרור הוא לאחר תאריך הקבלה של חולה למחלקה או שאין תאריך שחרור

```sql
ALTER TABLE Patients
ADD CONSTRAINT chk_release_date CHECK (ReleaseDate >= AdmissionDate OR ReleaseDate IS NULL);
```

```sql
INSERT INTO Patients (PatientID, FirstName, LastName, DateOfBirth, Gender, Phone, AdmissionDate, ReleaseDate, Address, DepartmentID)
VALUES (1, 'John', 'Doe', TO_DATE('1990-01-01', 'YYYY-MM-DD'), 'M', '050-1234567', TO_DATE('2023-08-01', 'YYYY-MM-DD'), TO_DATE('2023-07-01', 'YYYY-MM-DD'),
```



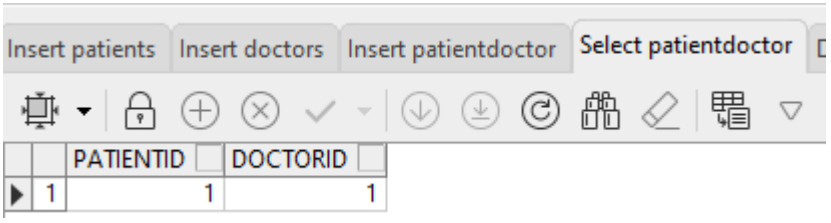
7. הוספה שמחיקת מטופל תגרור את מחיקת הרשומות טיפול שלו

```sql
ALTER TABLE PatientDoctor
DROP CONSTRAINT fk_patient;

ALTER TABLE PatientDoctor
ADD CONSTRAINT fk_patient FOREIGN KEY (PatientID)
REFERENCES Patients(PatientID) ON DELETE CASCADE;
```

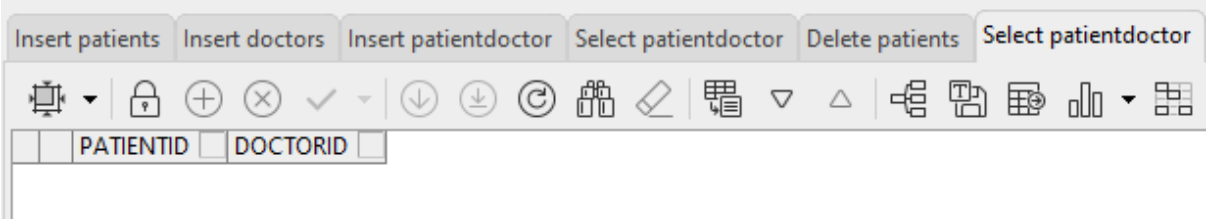
שאילתא שתדמה את זה:

```sql
-- Insert sample patient and doctor data
INSERT INTO Patients (PatientID, FirstName, LastName, DateOfBirth, Gender, Phone, AdmissionDate, ReleaseDate, Address, DepartmentID)
VALUES (1, 'John', 'Doe', TO_DATE('1980-01-01', 'YYYY-MM-DD'), 'M', '555-1234', SYSDATE, NULL, '123 Street', 1);

INSERT INTO Doctors (DoctorID, FirstName, LastName, Specialty, Phone, DateOfBirth, HireDate, Salary, DistanceFromHospital, DepartmentID)
VALUES (1, 'Jane', 'Smith', 'Cardiology', '555-5678', TO_DATE('1975-01-01', 'YYYY-MM-DD'), TO_DATE('2000-01-01', 'YYYY-MM-DD'), 10000, 15, 1);

-- Link the patient to the doctor in the PatientDoctor table
INSERT INTO PatientDoctor (PatientID, DoctorID)
VALUES (1, 1);

SELECT * FROM PatientDoctor WHERE PatientID = 1;

-- Delete the patient and verify if related records in PatientDoctor are deleted
DELETE FROM Patients WHERE PatientID = 1;

-- Check if the related records in PatientDoctor are deleted
SELECT * FROM PatientDoctor WHERE PatientID = 1;

ROLLBACK;
```

לפני המחיקה



לאחר המחיקה



שלב ג'

ניתוח ביצועים של רופאים

יצרנו מערכת שמנתחת את ביצועי הרופאים במחלקות השונות. המערכת מחשבת אחוזונים של ביצועים, מעלה את השכר לרופאים מצטיינים, מסווגת את הרופאים בהתאם לתוצאותיהם, ומדווחת בצורה ברורה ומסודרת.

מטרות העבודה:

1. **חישוב ביצועי רופאים:** חישוב ממוצע משך הזמן שרופא מטפל בחולה בהתבסס על תאריכי האשפוז והשחרור של החולים ביחס לכמות החולים שטיפל.
2. **חלוקה לאחוזונים:** סיווג הרופאים לאחוזוני ביצועים (Top 10%, Top 20%, Top 50%, ועוד), תוך מתן העלאות שכר למצטיינים ומתן חיווי למי שדורש שיפור.
3. **מודולריות:** בניית המערכת באופן מודולרי כדי לאפשר תחזוקה והרחבה קלים יותר.
4. **טיפול בחריגות:** התמודדות עם חריגות באופן ממוקד בכל פונקציה או פרוצדורה כדי להבטיח תהליך חלק וללא שגיאות בלתי צפויות.

מהלך העבודה:

במהלך העבודה פיתחנו מספר פונקציות ופרוצדורות, שכל אחת מהן מבצעת חלק מוגדר מהמשימה:

פונקציה לשליפת מחלקות (`get_departments`): פונקציה זו אחראית לשליפת רשימת המחלקות מבסיס הנתונים באמצעות קורסור.

```
CREATE OR REPLACE FUNCTION get_departments
RETURN SYS_REFCURSOR
IS
    department_cursor SYS_REFCURSOR;
BEGIN
    -- Open a cursor to fetch department list
    OPEN department_cursor FOR
        SELECT DepartmentID, DepartmentName
        FROM Departments;
    RETURN department_cursor;

EXCEPTION
    WHEN NO_DATA_FOUND THEN
        DBMS_OUTPUT.PUT_LINE('Error: No departments found.');
    WHEN OTHERS THEN
        DBMS_OUTPUT.PUT_LINE('Error: Unable to fetch departments - ' || SQLERRM);
END;
```

פרוצדורה לעיבוד מחלקה (`process_department`): פרוצדורה זו מקבלת את מזהה המחלקה ומעבדת אותה, כולל חישוב אחוזונים והדפסת נתוני הרופאים.

```oracle
CREATE OR REPLACE PROCEDURE process_department(department_id IN NUMBER, department_name IN VARCHAR2)
IS
    top_10_threshold NUMBER;
    top_20_threshold NUMBER;
    top_50_threshold NUMBER;
    low_20_threshold NUMBER;
    percentiles_cursor SYS_REFCURSOR;
BEGIN
    -- Print department name and ID
    DBMS_OUTPUT.PUT_LINE(CHR(10) || CHR(10) || 'Department: ' || department_name || ' (ID: ' || department_id || ')');

    -- Fetch percentiles for the department
    percentiles_cursor := get_department_percentiles(department_id);
    FETCH percentiles_cursor INTO top_10_threshold, top_20_threshold, top_50_threshold, low_20_threshold;

    -- Print calculated percentiles
    DBMS_OUTPUT.PUT_LINE('Percentiles calculated: Top 10%: ' || top_10_threshold ||
                         ', Top 20%: ' || top_20_threshold ||
                         ', Top 50%: ' || top_50_threshold ||
                         ', Bottom 20%: ' || low_20_threshold);

    -- Print doctor performance for the department
    print_doctor_performance(department_id);

EXCEPTION
    WHEN NO_DATA_FOUND THEN
        DBMS_OUTPUT.PUT_LINE('Error: No data found for department ' || department_name);
    WHEN OTHERS THEN
        DBMS_OUTPUT.PUT_LINE('Error processing department ' || department_name || ' - ' || SQLERRM);
END;
```

פונקציה לחישוב אחוזונים (`get_department_percentiles`): פונקציה זו מחשבת את האחוזונים השונים על בסיס ביצועי הרופאים בכל מחלקה.

```oracle
CREATE OR REPLACE FUNCTION get_department_percentiles(department_id IN NUMBER)
RETURN SYS_REFCURSOR
IS
    percentiles_cursor SYS_REFCURSOR;
BEGIN
    -- Open cursor for percentile calculations
    OPEN percentiles_cursor FOR
    SELECT ROUND(PERCENTILE_CONT(0.9) WITHIN GROUP (ORDER BY performance_score), 2) AS top_10,
           ROUND(PERCENTILE_CONT(0.8) WITHIN GROUP (ORDER BY performance_score), 2) AS top_20,
           ROUND(PERCENTILE_CONT(0.5) WITHIN GROUP (ORDER BY performance_score), 2) AS top_50,
           ROUND(PERCENTILE_CONT(0.2) WITHIN GROUP (ORDER BY performance_score), 2) AS low_20
    FROM (
        SELECT d.DoctorID,
               AVG(NVL(p.ReleaseDate, SYSDATE) - p.AdmissionDate) / COUNT(p.PatientID) AS performance_score
        FROM Doctors d
        LEFT JOIN PatientDoctor pd ON d.DoctorID = pd.DoctorID
        LEFT JOIN Patients p ON pd.PatientID = p.PatientID
        WHERE d.DepartmentID = department_id
        GROUP BY d.DoctorID
    );

    RETURN percentiles_cursor;

EXCEPTION
    WHEN NO_DATA_FOUND THEN
        DBMS_OUTPUT.PUT_LINE('Error: No performance data found for department ' || department_id);
    WHEN OTHERS THEN
        DBMS_OUTPUT.PUT_LINE('Error calculating percentiles for department ' || department_id || ' - ' || SQLERRM);
        RETURN NULL;
END;
```

פרוצדורה להדפסת ביצועי רופאים (`print_doctor_performance`): פרוצדורה זו מציגה את הביצועים של הרופאים לפי הקטגוריות שהוגדרו (מצטיינים, בסדר, ודורשים שיפור).

```oracle
-- Procedure to print doctor performance for a given department
CREATE OR REPLACE PROCEDURE print_doctor_performance(department_id IN NUMBER)
IS
    performance_cursor SYS_REFCURSOR;
    percentiles_cursor SYS_REFCURSOR;
    doctor_id NUMBER;
    doctor_name VARCHAR2(100);
    performance_score NUMBER;
    top_10_threshold NUMBER;
    top_20_threshold NUMBER;
    top_50_threshold NUMBER;
    low_20_threshold NUMBER;
    top_10_doctors VARCHAR2(2000) := '';
    top_20_doctors VARCHAR2(2000) := '';
    top_50_doctors VARCHAR2(2000) := '';
    low_20_doctors VARCHAR2(2000) := '';
    improvement_doctors VARCHAR2(2000) := '';
BEGIN
    -- Fetch percentiles from the external function
    percentiles_cursor := get_department_percentiles(department_id);
    FETCH percentiles_cursor INTO top_10_threshold, top_20_threshold, top_50_threshold, low_20_threshold;

    -- Fetch doctor performance sorted by score
    OPEN performance_cursor FOR
        SELECT d.DoctorID, d.FirstName || ' ' || d.LastName AS DoctorName,
               AVG(NVL(p.ReleaseDate, SYSDATE) - p.AdmissionDate) / COUNT(p.PatientID) AS performance_score
        FROM Doctors d
        LEFT JOIN PatientDoctor pd ON d.DoctorID = pd.DoctorID
        LEFT JOIN Patients p ON pd.PatientID = p.PatientID
        WHERE d.DepartmentID = department_id
        GROUP BY d.DoctorID, d.FirstName, d.LastName
        ORDER BY performance_score DESC;

    -- Loop through doctors and categorize based on performance
    LOOP
        FETCH performance_cursor INTO doctor_id, doctor_name, performance_score;
        EXIT WHEN performance_cursor%NOTFOUND;

        -- Round performance score
        performance_score := ROUND(performance_score, 2);

        -- Add doctor to the appropriate category
        IF performance_score >= top_10_threshold THEN
            top_10_doctors := top_10_doctors || CHR(10) || '  Doctor ' || doctor_name || ' (ID: ' || doctor_id || ') - Performance Score: ' || performance_score;
        ELSIF performance_score >= top_20_threshold THEN
            top_20_doctors := top_20_doctors || CHR(10) || '  Doctor ' || doctor_name || ' (ID: ' || doctor_id || ') - Performance Score: ' || performance_score;
        ELSIF performance_score >= top_50_threshold THEN
            top_50_doctors := top_50_doctors || CHR(10) || '  Doctor ' || doctor_name || ' (ID: ' || doctor_id || ') - Performance Score: ' || performance_score;
        ELSIF performance_score >= low_20_threshold THEN
            low_20_doctors := low_20_doctors || CHR(10) || '  Doctor ' || doctor_name || ' (ID: ' || doctor_id || ') - Performance Score: ' || performance_score;
        ELSE
            improvement_doctors := improvement_doctors || CHR(10) || '  Doctor ' || doctor_name || ' (ID: ' || doctor_id || ') - Performance Score: ' || performance_score;
        END IF;
    END LOOP;

    -- Print results for each category
    IF top_10_doctors IS NOT NULL THEN
        DBMS_OUTPUT.PUT_LINE(CHR(10) || 'Top 10% doctors with a salary raise of 1000: ' || top_10_doctors);
    END IF;

    IF top_20_doctors IS NOT NULL THEN
        DBMS_OUTPUT.PUT_LINE(CHR(10) || 'Top 20% doctors with a salary raise of 500: ' || top_20_doctors);
    END IF;

    IF top_50_doctors IS NOT NULL THEN
        DBMS_OUTPUT.PUT_LINE(CHR(10) || 'Top 50% doctors with a rating of "Very Good": ' || top_50_doctors);
    END IF;

    IF low_20_doctors IS NOT NULL THEN
        DBMS_OUTPUT.PUT_LINE(CHR(10) || 'Doctors with a rating of "Almost Good": ' || low_20_doctors);
    END IF;

    IF improvement_doctors IS NOT NULL THEN
        DBMS_OUTPUT.PUT_LINE(CHR(10) || 'Doctors needing improvement (Bottom 20%): ' || improvement_doctors);
    END IF;

    -- Close the performance cursor
    CLOSE performance_cursor;
```

```
    EXCEPTION
    WHEN NO_DATA_FOUND THEN
        DBMS_OUTPUT.PUT_LINE('Error: No doctor performance data found for department ' || department_id);
    WHEN OTHERS THEN
        DBMS_OUTPUT.PUT_LINE('Error printing doctor performance for department ' || department_id || ' - ' || SQLERRM);
        CLOSE performance_cursor;
```

פרוצדורה לטיפול כללי בשגיאות (`handle_general_error`): פרוצדורה זו מטפלת בשגיאות כלליות שלא נלכדו בפרוצדורות האחרות.

```
-- Procedure to handle errors and exceptions
CREATE OR REPLACE PROCEDURE handle_general_error(error_code IN NUMBER, error_message IN VARCHAR2)
IS
BEGIN
    -- Print the error code and message
    DBMS_OUTPUT.PUT_LINE('Error Code: ' || error_code);
    DBMS_OUTPUT.PUT_LINE('Error Message: ' || error_message);
    -- Rollback the transaction if needed
    ROLLBACK;
END;
```

דוגמת הרצה:

```
Department: Pediatrics (ID: 1)
Percentiles calculated: Top 10%: 162.49, Top 20%: 132.45, Top 50%: 70.28, Bottom 20%: 27

Top 10% doctors with a salary raise of 1000:
  Doctor Linda Williams (ID: 309043170) - Performance Score: 276.6
  Doctor Alice Smith (ID: 304078104) - Performance Score: 236.55
  Doctor James Jones (ID: 207136636) - Performance Score: 169.99

Top 20% doctors with a salary raise of 500:
  Doctor Linda Taylor (ID: 206367479) - Performance Score: 160.62
  Doctor David Smith (ID: 207448768) - Performance Score: 156.24
  Doctor James Jones (ID: 301135367) - Performance Score: 150.38

Top 50% doctors with a rating of "Very Good":
  Doctor Alice Williams (ID: 200096911) - Performance Score: 120.5
  Doctor Sara Williams (ID: 208771808) - Performance Score: 111.09
  Doctor Alice Wilson (ID: 304942290) - Performance Score: 107.66
  Doctor Jane Taylor (ID: 304539706) - Performance Score: 104.76
  Doctor John Jones (ID: 200498237) - Performance Score: 97.95
  Doctor James Moore (ID: 305317262) - Performance Score: 91.13
  Doctor John Moore (ID: 306780302) - Performance Score: 89.56
  Doctor David Jones (ID: 307401220) - Performance Score: 87.97
  Doctor James Brown (ID: 208132669) - Performance Score: 70.28

Doctors with a rating of "Almost Good":
  Doctor Robert Davis (ID: 207328765) - Performance Score: 66.29
  Doctor Jane Taylor (ID: 203022445) - Performance Score: 56.65
  Doctor Sara Williams (ID: 202537975) - Performance Score: 47.7
  Doctor Jane Smith (ID: 202876381) - Performance Score: 39.06
  Doctor Michael Johnson (ID: 300417897) - Performance Score: 37.85
```

דוגמה לשינוי בבסיס הנתונים לרופאים שקיבלו העלאה עבור מחלקה 1:

לאחר הריצה

	DOCTORID	FIRSTNAME	LASTNAME	SALARY
1	200498237	John	Jones	138064
2	304942290	Alice	Wilson	80317
3	204350645	Alice	Taylor	94645
4	206367479	Linda	Taylor	171123
5	209929760	Michael	Smith	140521
6	207136636	James	Jones	57611
7	307401220	David	Jones	67109
8	208771808	Sara	Williams	177472
9	204901193	Linda	Moore	75839
10	300417897	Michael	Johnson	175967
11	208132669	James	Brown	160632
12	209531628	Linda	Moore	57191
13	309043170	Linda	Williams	74364
14	309815512	Michael	Moore	84983
15	207328765	Robert	Davis	133693
16	306780302	John	Moore	182203
17	305317262	James	Moore	167570
18	309925512	David	Wilson	59893
19	203022445	Jane	Taylor	90835
20	200096911	Alice	Williams	58587
21	304539706	Jane	Taylor	155402
22	202876381	Jane	Smith	180792
23	301135367	James	Jones	183956
24	304078104	Alice	Smith	200447
25	202537975	Sara	Williams	156153
26	307844344	Jane	Wilson	141892
27	302232987	Linda	Williams	111419
28	207448768	David	Smith	176372
29	302291201	Sara	Johnson	191253

לפני הריצה

	DOCTORID	FIRSTNAME	LASTNAME	SALARY
1	200498237	John	Jones	138064
2	304942290	Alice	Wilson	80317
3	204350645	Alice	Taylor	94645
4	206367479	Linda	Taylor	170623
5	209929760	Michael	Smith	140521
6	207136636	James	Jones	56611
7	307401220	David	Jones	67109
8	208771808	Sara	Williams	177472
9	204901193	Linda	Moore	75839
10	300417897	Michael	Johnson	175967
11	208132669	James	Brown	160632
12	209531628	Linda	Moore	57191
13	309043170	Linda	Williams	73364
14	309815512	Michael	Moore	84983
15	207328765	Robert	Davis	133693
16	306780302	John	Moore	182203
17	305317262	James	Moore	167570
18	309925512	David	Wilson	59893
19	203022445	Jane	Taylor	90835
20	200096911	Alice	Williams	58587
21	304539706	Jane	Taylor	155402
22	202876381	Jane	Smith	180792
23	301135367	James	Jones	183456
24	304078104	Alice	Smith	199447
25	202537975	Sara	Williams	156153
26	307844344	Jane	Wilson	141892
27	302232987	Linda	Williams	111419
28	207448768	David	Smith	175872
29	302291201	Sara	Johnson	191253