Narodowa baza danych Covid-19

Michał Kuliński Jakub Małkowski

1 Podstawowe założenia projektu

1.1 Cel i główne założenia projektu

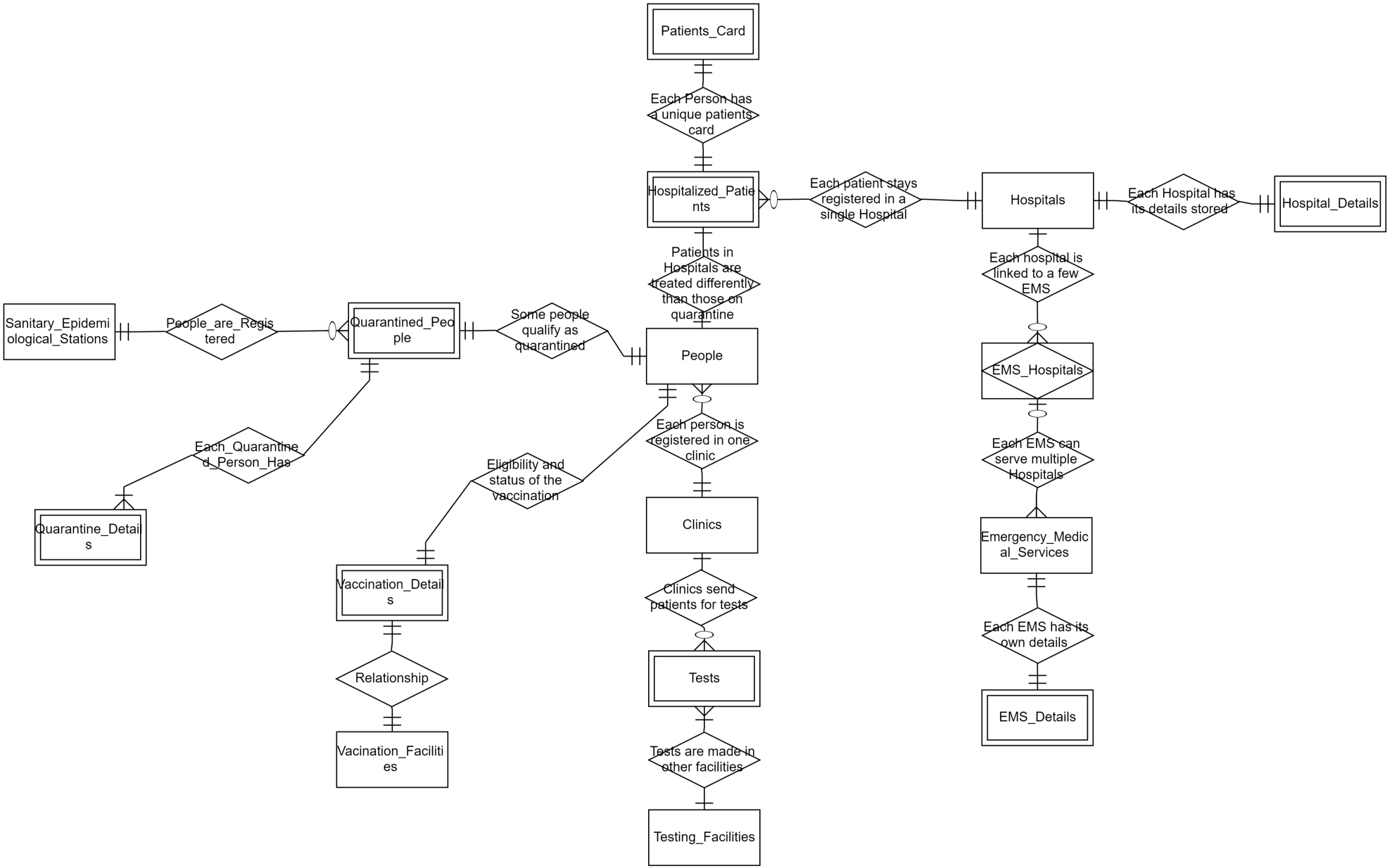
Celem bazy danych jest umożliwienie przechowywania kluczowych i najważniejszych informacji dotyczących ludzi dotkniętych wirusem oraz systemu z tym związanym. W bazie danych przechowywane są informacje o kwarantannach, szczepieniach, ludziach aktualnie hospitalizowanych oraz przeprowadzanych testach.
Ponadto w bazie danych znajdują się informacje o przychodniach, szpitalach i pogotowiu ratunkowym. Przechowywane są również informacje o stacjach sanitarno-epidemiologicznych, punktach szczepień i punktach testowania.

1.2 Możliwości

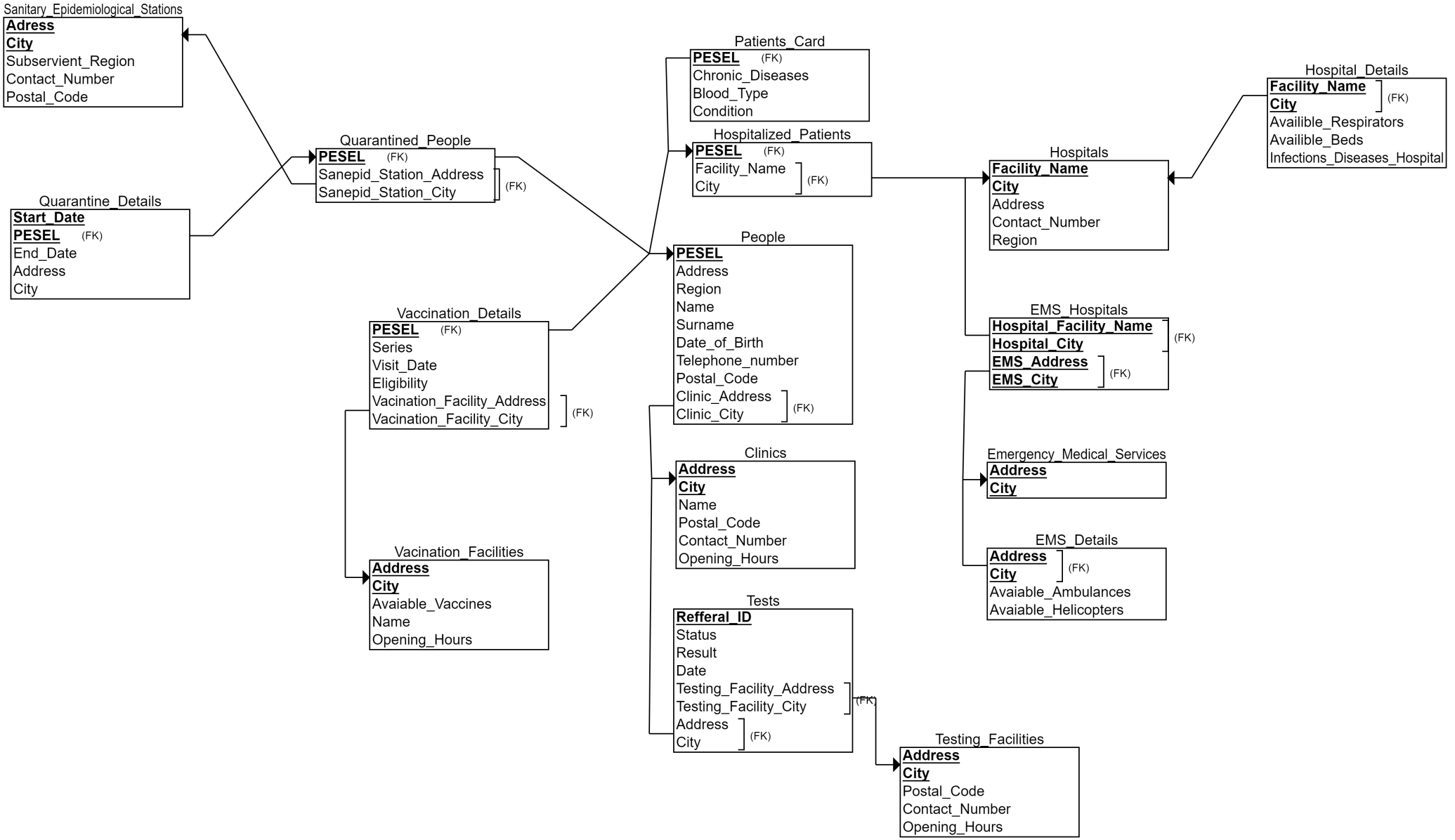
1.2.1 Ludzie

Tabela "People" jest centralnym punktem bazy danych, do którego można dodawać ludzi za pomocą procedury składowanej "AddPerson", która jednocześnie przypisuje im przychodnię oraz punkt szczepień.

2 Diagram ER


Patients_Card
Each Person has a unique patients card
Hospitalized_Patients
Each patient stays registered in a single Hospital
Hospitals
Each Hospital has its details stored
Hospital_Details
Patients in Hospitals are treated differently than those on quarantine
Each hospital is linked to a few EMS
Sanitary_Epidemiological_Stations
People_are_Registered
Quarantined_People
Some people qualify as quarantined
People
EMS_Hospitals
Each_Quarantined_Person_Has
Each person is registered in one clinic
Each EMS can serve multiple Hospitals
Quarantine_Details
Eligibility and status of the vaccination
Clinics
Emergency_Medical_Services
Vaccination_Details
Clinics send patients for tests
Each EMS has its own details
Relationship
Tests
EMS_Details
Vaccination_Facilities
Tests are made in other facilities
Testing_Facilities

3 Schemat bazy danych



4 Widoki

4.1 Vaccination Eligibility

Widok przedstawiający informacje na temat kwalifikowalności do szczepienia.

```
01 |  DROP VIEW IF EXISTS dbo.Eligibility;
02 |  GO
03 |
04 |  CREATE VIEW Eligibility AS
05 |          SELECT Name, Surname, V.Eligibility From People P
06 |          INNER JOIN Vaccination_Details V ON V.PESEL = P.PESEL;
07 |  GO
```

4.2 Hospital-EMS connections

Widok przedstawiający informacje na temat współpracujących szpitali i pogotowia ratunkowego.

```
01 |  DROP VIEW IF EXISTS dbo.Hospital_EMS_Connections;
02 |  GO
03 |
04 |  CREATE VIEW Hospital_EMS_Connections
05 |  AS
06 |          SELECT H.Facility_Name, H.City, E.EMS_Address AS EMS_Address From Hospitals H
07 |          INNER JOIN EMS_Hospitals E ON E.Hospital_Facility_Name = H.Facility_Name AND E.Hospital_City = H.City
08 |          INNER JOIN EMS_Details D ON D.Address = E.EMS_Address AND D.City = E.EMS_City
09 |  GO
```

4.3 Hospitals in region

Widok przedstawiający informacje na temat szpitali w regionie oraz aktualny stan respiratorów i łóżek dla każdego z nich.

```
01 |  DROP VIEW IF EXISTS dbo.HospitalsInRegion;
02 |  GO
03 |
04 |  CREATE VIEW HospitalsInRegion AS
05 |          SELECT H.Facility_Name, H.Address, HD.Available_Beds, Hd.Available_Respirators, HD.Infections_Diseases_Hospital
06 |          FROM Hospitals H
07 |          JOIN Hospital_Details HD ON HD.Facility_Name = H.Facility_Name
08 |          ORDER BY H.Region DESC;
09 |  GO
```

4.4 Infections per region

Widok przedstawiający liczbę zakażeń na każdy region.

```
01 |  DROP VIEW IF EXISTS dbo.Infections_Per_Region;
02 |  GO
03 |
04 |  CREATE VIEW Infections_Per_Region AS
05 |          SELECT DISTINCT P.Region, (SELECT DISTINCT COUNT(T.PESEL) FROM Tests T
06 |                  GROUP BY T.Result HAVING T.Result='Positive') Infections FROM People P;
07 |  GO
```

4.5 Patients in hospital

Widok przedstawiający liczbę pacjentów na każdy szpital.

```
01 |  DROP VIEW IF EXISTS dbo.Patients_in_Hospital;
02 |  GO
03 |
04 |  CREATE VIEW Patients_in_Hospital AS
05 |          SELECT H.Facility_Name, H.Address, (SELECT Count(*) FROM Hospitalized_Patients HP
06 |          GROUP BY HP.Facility_Name
07 |          HAVING HP.Facility_Name = H.Facility_Name) AS Patients_Per_Hospital FROM Hospitals H;
08 |  GO
```

5 Procedury składowane

5.1 Add Clinic

Procedura pozwalająca dodać nową przychodnię do tabeli *Clinics*.

```
01 |  DROP PROC IF EXISTS dbo.AddClinic;
02 |  GO
03 |
04 |  CREATE PROC dbo.AddClinic
05 |
06 |  @Address VARCHAR(50),
07 |  @City VARCHAR(30),
08 |  @Name VARCHAR(50),
09 |  @Postal_Code VARCHAR(6),
10 |  @Contact_Number VARCHAR(20)
11 |
12 |  AS
13 |
14 |  INSERT INTO Clinics(Address, City, Name, Postal_Code, Contact_Number)
15 |  VALUES(@Address, @City, @Name, @Postal_Code, @Contact_Number)
16 |
17 |  GO
```

5.2 Add Person

Procedura pozwalająca dodać nową osobę do tabeli *People*. Dodatkowo zostaje jej przypisany aktualny status dotyczący szczepienia w tabeli *Vaccination Details*.

```
01 |  DROP PROC IF EXISTS dbo.AddPerson;
02 |  GO
03 |
04 |  CREATE PROC dbo.AddPerson
05 |
06 |  @Address VARCHAR(50),
07 |  @PESEL VARCHAR(11),
08 |  @Region VARCHAR(30),
09 |  @Name VARCHAR(30),
10 |  @Surname VARCHAR(30),
11 |  @Date_of_Birth DATE,
12 |  @Telephone_number VARCHAR(20),
13 |  @Postal_Code VARCHAR(6),
14 |  @Clinic_Address VARCHAR(50),
15 |  @Clinic_City VARCHAR(30),
16 |  @Vaccination_Facility_Address VARCHAR(50),
17 |  @Vaccination_Facility_City VARCHAR(30)
18 |
19 |  AS
20 |
21 |  INSERT INTO People(Address, PESEL, Region, Name, Surname, Date_of_Birth, Telephone_number, Postal_Code, Clinic_Address, Clinic_City)
22 |  VALUES(@Address, @PESEL, @Region, @Name, @Surname, @Date_of_Birth, @Telephone_number, @Postal_Code, @Clinic_Address, @Clinic_City)
23 |
24 |  INSERT INTO Vaccination_Details(PESEL, Series, Visit_Date, Eligibility, Vaccination_Facility_Address, Vaccination_Facility_City)
25 |  VALUES(@PESEL, NULL, NULL, 0, @Vaccination_Facility_Address, @Vaccination_Facility_City)
26 |
27 |  GO
```

5.3 Add Sanepid

Procedura pozwalająca dodać nowy punkt sanitarno-epidemiologiczny do tabeli *Sanitary Epidemiological Stations*.

```
01 |  DROP PROC IF EXISTS dbo.AddSanepid
02 |  GO
03 |
04 |  CREATE PROC dbo.AddSanepid
05 |
06 |  @Address VARCHAR(50),
07 |  @Subservient_Region VARCHAR(30),
08 |  @Contact_Number VARCHAR(20),
09 |  @City VARCHAR(30),
10 |  @Postal_Code VARCHAR(6)
11 |
12 |  AS
13 |
14 |  INSERT Sanitary_Epidemiological_Stations(Address, Subservient_Region, Contact_Number, City, Postal_Code)
15 |  VALUES(@Address, @Subservient_Region, @Contact_Number, @City, @Postal_Code)
16 |
17 |  GO
```

5.4 Add Testing Facility

Procedura pozwalająca dodać nowy punkt przeprowadzania testów do tabeli *Testing Facilities*.

```
01 |  DROP PROC IF EXISTS dbo.AddTestingFacility
02 |  GO
03 |
04 |  CREATE PROC dbo.AddTestingFacility
05 |
06 |  @Address VARCHAR(50),
07 |  @City VARCHAR(30),
08 |  @Postal_Code VARCHAR(6),
09 |  @Contact_Number INT
10 |
11 |  AS
12 |
13 |  INSERT INTO Testing_Facilities(Address, City, Postal_Code, Contact_Number)
14 |  VALUES(@Address, @City, @Postal_Code, @Contact_Number)
15 |
16 |  GO
```

5.5 Add Vaccination Facility

Procedura pozwalająca dodać nowy punkt przeprowadzania szczepień do tabeli *Vaccination Facilities*.

```
01 |  DROP PROC IF EXISTS dbo.AddVaccinationFacility;
02 |  GO
03 |
04 |  CREATE PROC dbo.AddVaccinationFacility
05 |
06 |  @Available_Vaccines INT,
07 |  @Address VARCHAR(50),
08 |  @City VARCHAR(30),
09 |  @Name VARCHAR(50)
10 |
11 |  AS
12 |
13 |  INSERT INTO Vaccination_Facilities(Available_Vaccines, Address, City, Name)
14 |  VALUES(@Available_Vaccines, @Address, @City, @Name)
15 |
16 |  GO
```

5.6 Make Eligible For Vaccination

Procedura pozwalająca na zmianę kwalifikowalności do szczepienia poprzez podanie numeru PESEL w tabeli *Vaccination Details*.

```
01 |  DROP PROC IF EXISTS dbo.MakeEligibleForVaccination
02 |  GO
03 |
04 |  CREATE PROC dbo.MakeEligibleForVaccination
05 |
06 |  @PESEL VARCHAR(11)
07 |
08 |  AS
09 |
10 |  UPDATE Vaccination_Details
11 |  SET Eligibility = 1
12 |  WHERE PESEL = @PESEL
13 |
14 |  GO
```

5.7 Quarantine

Procedura pozwalająca na poddanie osoby kwarantannie poprzez dodanie osoby do tabeli *Quarantined People* oraz dodanie szczegół dotyczących kwarantanny do tabeli *Quarantine Details*.

```
01 |  DROP PROC IF EXISTS dbo.Quarantine
02 |  GO
03 |
04 |  CREATE PROC dbo.Quarantine
05 |
06 |  @Start_Date DATE,
07 |  @End_Date DATE,
08 |  @Address VARCHAR(50),
09 |  @City VARCHAR(30),
10 |  @PESEL VARCHAR(11),
11 |  @Sanepid_Station_Address VARCHAR(50),
12 |  @Sanepid_Station_City VARCHAR(30)
13 |
14 |  AS
15 |
16 |  INSERT Quarantined_People(PESEL, Sanepid_Station_Address, Sanepid_Station_City)
17 |  VALUES(@PESEL, @Sanepid_Station_Address, @Sanepid_Station_City)
18 |
19 |  INSERT Quarantine_Details(Start_Date, End_Date, Address, City, PESEL)
20 |  VALUES(@Start_Date, @End_Date, @Address, @City, @PESEL)
21 |
22 |  GO
```

5.8 Add Patient Card

Procedura pozwalająca na dodanie karty pacjenta do tabeli *Patient Cards*.

```
01 |  DROP PROC IF EXISTS dbo.AddPatientCard;
02 |  GO
03 |
04 |  CREATE PROC dbo.AddPatientCard
05 |
06 |  @Chronic_Diseases VARCHAR(100),
07 |  @Blood_Type VARCHAR(10),
08 |  @Condition VARCHAR(30),
09 |  @PESEL VARCHAR(11)
10 |
11 |  AS
12 |
13 |  INSERT INTO Patients_Card(Chronic_Diseases, Blood_Type, Condition, PESEL)
14 |  VALUES(@Chronic_Diseases, @Blood_Type, @Condition, @PESEL)
15 |
16 |  GO
```

5.9 Add Patient

Procedura pozwalająca na dodanie pacjenta do tabeli *Patients*.

```
01 |  DROP PROC IF EXISTS dbo.AddPatient;
02 |  GO
03 |
04 |  CREATE PROC dbo.AddPatient
05 |
06 |  @PESEL VARCHAR(11)
07 |
08 |  AS
09 |
10 |  INSERT INTO dbo.Hospitalized_Patients(PESEL, Facility_Name, City)
11 |  SELECT TOP 1 @PESEL, H.Facility_Name, H.City FROM dbo.AVAILABLE_HOSPITALS(@PESEL) H
12 |  WHERE Available_Beds > 0
13 |  ORDER BY NEWID()
14 |  GO
```

5.10 Discharge Patient

Procedura pozwalająca na wypis pacjenta i usunięcie go z tabeli *Hospitalized Patients*.

```
01 |  DROP PROC IF EXISTS dbo.DischargePatient;
02 |  GO
03 |
04 |  CREATE PROC dbo.DischargePatient
05 |
06 |  @PESEL VARCHAR(11)
07 |
08 |  AS
09 |
10 |  BEGIN TRANSACTION
11 |          DELETE FROM Hospitalized_Patients WHERE PESEL = @PESEL
12 |  COMMIT;
```

6 Funkcje

6.1 Active assigned quarantines

Funkcja zwraca wszystkie bieżące kwarantanny zarządzone przez daną jednostkę sanepidu.

```
01 |  DROP FUNCTION IF EXISTS [dbo].[ACTIVE_SANEPID_ASSIGNED_QUARANTINES];
02 |  GO
03 |
04 |  CREATE FUNCTION ACTIVE_SANEPID_ASSIGNED_QUARANTINES (@SNPD_A varchar,@SNPD_C varchar)
05 |  RETURNS TABLE
06 |  AS
07 |  RETURN(
08 |          SELECT QD.PESEL,QD.Start_Date,QD.End_Date FROM Sanitary_Epidemiological_Stations T
09 |          JOIN Quarantined_People QP ON T.Address=QP.Sanepid_Station_Address AND T.City=QP.Sanepid_Station_City
10 |          JOIN Quarantine_Details QD ON QP.PESEL=QD.PESEL
11 |          WHERE T.Address = @SNPD_A AND T.City = @SNPD_C AND QD.End_Date<GETDATE()
12 |  )
13 |  GO
```

6.2 Available beds

Funkcja zwraca ilość dostępnych łóżek dla danego szpitala, wykorzystywana m.in. w funkcji *Available Hospitals*.

```
01 |  DROP FUNCTION IF EXISTS [dbo].[Available_Beds];
02 |  GO
03 |
04 |  CREATE FUNCTION Available_Beds(@HSPN varchar,@HSPC varchar)
05 |  RETURNS int
06 |  AS
07 |  BEGIN
08 |          DECLARE @BDS int;
09 |          SET @BDS = (SELECT HD.Available_Beds FROM Hospital_Details HD WHERE HD.Facility_Name=@HSPN AND HD.City=@HSPC)
10 |          RETURN @BDS;
11 |  END;
```

6.3 Available hospitals

Funkcja zwraca tabelę szpitali do których może zostać przyjęty pacjent biorąc pod uwagę region oraz liczbę dostępnych łóżek, wykorzystywana m.in. w procedurze składowanej *Add Patient*.

```
01 |  DROP FUNCTION IF EXISTS [dbo].[Available_Hospitals];
02 |  GO
03 |
04 |  CREATE FUNCTION Available_Hospitals(@PESEL varchar)
05 |          RETURNS TABLE
06 |          AS
07 |          RETURN(
08 |          SELECT H.Facility_Name,H.City,H.Address, H.Contact_Number FROM Hospitals H
09 |          INNER JOIN People P ON H.Region=P.Region AND H.Region=P.Region
10 |          WHERE P.PESEL=@PESEL AND dbo.Available_Beds(H.Address,H.City) > 0
11 |  )
12 |  GO
```

6.4 Check test result

Funkcja sprawdzająca, czy pacjent był testowany, a jeżeli tak, jaki był wynik testu.

```
01 |  DROP FUNCTION IF EXISTS [dbo].[Check_Test_Result];
02 |  GO
03 |
04 |  CREATE FUNCTION Check_Test_Result(@PESEL varchar)
05 |  RETURNS Varchar
06 |  AS
07 |  BEGIN
08 |          DECLARE @TST varchar;
09 |          IF (SELECT MAX(T.Test_Date) FROM Tests T WHERE T.Refferal_ID=@PESEL) IS NULL
10 |                  SET @TST = 'Pacjent nie byl testowany'
11 |          ELSE
12 |                  SET @TST = (SELECT TE.Result FROM Tests TE WHERE TE.Test_Date=(SELECT MAX(T.Test_Date) FROM Tests T WHERE T.Refferal_ID=@PESEL))
13 |          RETURN @TST;
14 |  END;
```

6.5 Patient's contact number

Funkcja zwracająca numer do pacjenta z podanym peselem.

```
01 |  DROP FUNCTION IF EXISTS [dbo].[GetContactNumber];
02 |  GO
03 |
04 |  CREATE FUNCTION GetContactNumber(@PESEL varchar)
05 |  RETURNS Varchar
06 |  AS
07 |  BEGIN
08 |          DECLARE @TEL varchar;
09 |          SELECT @TEL = P.Telephone_number FROM People P WHERE @PESEL = P.PESEL
10 |          RETURN @TEL;
11 |  END;
```

7 Wyzwalacze

7.1 Create quarantine details

Wyzwalacz automatycznie dodający szczegóły kwarantanny w *Quarantine Details* po poddaniu osoby kwarantannie poprzez dodanie jej do tabeli *Quarantined People*.

```
01 |  CREATE TRIGGER dbo.AutoCreateQuarantineDetails
02 |  ON Quarantined_People
03 |  AFTER INSERT
04 |  AS
05 |  BEGIN
06 |          INSERT INTO Quarantine_Details
07 |          SELECT GETDATE(), QP.PESEL, CAST(DATEADD(week, 2, GETDATE()) as DATE), P.Address, P.Clinic_City FROM inserted QP
08 |          JOIN People P ON QP.PESEL=P.PESEL
09 |  END
```

7.2 Generate patient card

Wyzwalacz automatycznie wywołujący procedurę dodania karty pacjenta z podanym peselem po dodaniu pacjenta do tabeli *Hospitalized Patients*.

```
01 |  DROP TRIGGER IF EXISTS dbo.AutoGeneratePatientsCard;
02 |
03 |  GO;
04 |  CREATE TRIGGER dbo.AutoGeneratePatientsCard
05 |  ON Hospitalized_Patients
06 |  AFTER INSERT
07 |  AS
08 |  BEGIN
09 |          EXEC dbo.AddPatientCard @PESEL = inserted.PESEL
10 |  END
```

7.3 Quarantine

Wyzwalacz automatycznie poddający osobę kwarantannie poprzez dodanie nowego rekordu do tabeli *Quarantined People* po uzyskaniu pozytywnego wyniku w tabeli *Tests*.

```
01 |  DROP TRIGGER IF EXISTS dbo.AutoQuarantine;
02 |  GO;
03 |
04 |  CREATE TRIGGER dbo.AutoQuarantine
05 |  ON Tests
06 |  AFTER INSERT
07 |  AS
08 |  BEGIN
09 |          DECLARE @TestVariable AS VARCHAR(30)
10 |          SELECT @TestVariable = [Result] FROM inserted
11 |          IF(@TestVariable = 'Positive')
12 |                  INSERT INTO Quarantined_People
13 |                  SELECT I.PESEL, P.Region , (SELECT TOP 1 SE.Address FROM Sanitary_Epidemiological_Stations ORDER BY NEWID()) FROM People P
14 |                  JOIN inserted I ON P.PESEL=I.PESEL
15 |                  JOIN Sanitary_Epidemiological_Stations SE ON SE.Subservient_Region=P.Region
16 |  END
```

7.4 Make bed available

Wyzwalacz automatycznie zwalniający łóżko w szpitalu po usunięciu pacjenta z tabeli *Hospitalized Patients*.

```
01 |  DROP TRIGGER IF EXISTS dbo.MakeBedAvailable;
02 |  GO;
03 |
04 |  CREATE TRIGGER dbo.MakeBedAvailable
05 |  ON Hospitalized_Patients
06 |  AFTER DELETE
07 |  AS
08 |  BEGIN
09 |          BEGIN TRANSACTION
10 |          DECLARE @Fac_name AS varchar(30)
11 |          Declare @Fac_city AS varchar(30)
12 |          SELECT @Fac_name = [Facility_Name] FROM inserted
13 |          SELECT @Fac_city = [City] FROM inserted
14 |          UPDATE Hospital_Details
15 |          SET Available_Beds=Available_Beds+1
16 |          WHERE Facility_Name=@Fac_city AND City=@Fac_city
17 |          COMMIT;
18 |  END
```

7.5 Occupy bed

Wyzwalacz automatycznie zajmujący łóżko w szpitalu po dodaniu pacjenta do tabeli *Hospitalized Patients*.

```
01 |  DROP TRIGGER IF EXISTS dbo.OccupyBed;
02 |  GO;
03 |
04 |  CREATE TRIGGER dbo.OccupyBed
05 |  ON Hospitalized_Patients
06 |  AFTER INSERT
07 |  AS
08 |  BEGIN
09 |          BEGIN TRANSACTION
10 |          DECLARE @Fac_name AS varchar(30)
11 |          Declare @Fac_city AS varchar(30)
12 |          SELECT @Fac_name = [Facility_Name] FROM inserted
13 |          SELECT @Fac_city = [City] FROM inserted
14 |          UPDATE Hospital_Details
15 |          SET Available_Beds=Available_Beds-1
16 |          WHERE Facility_Name=@Fac_city AND City=@Fac_city
17 |          COMMIT;
18 |  END
```