Politechnika Świętokrzyska w Kielcach

Wydział Elektrotechniki, Automatyki i Informatyki

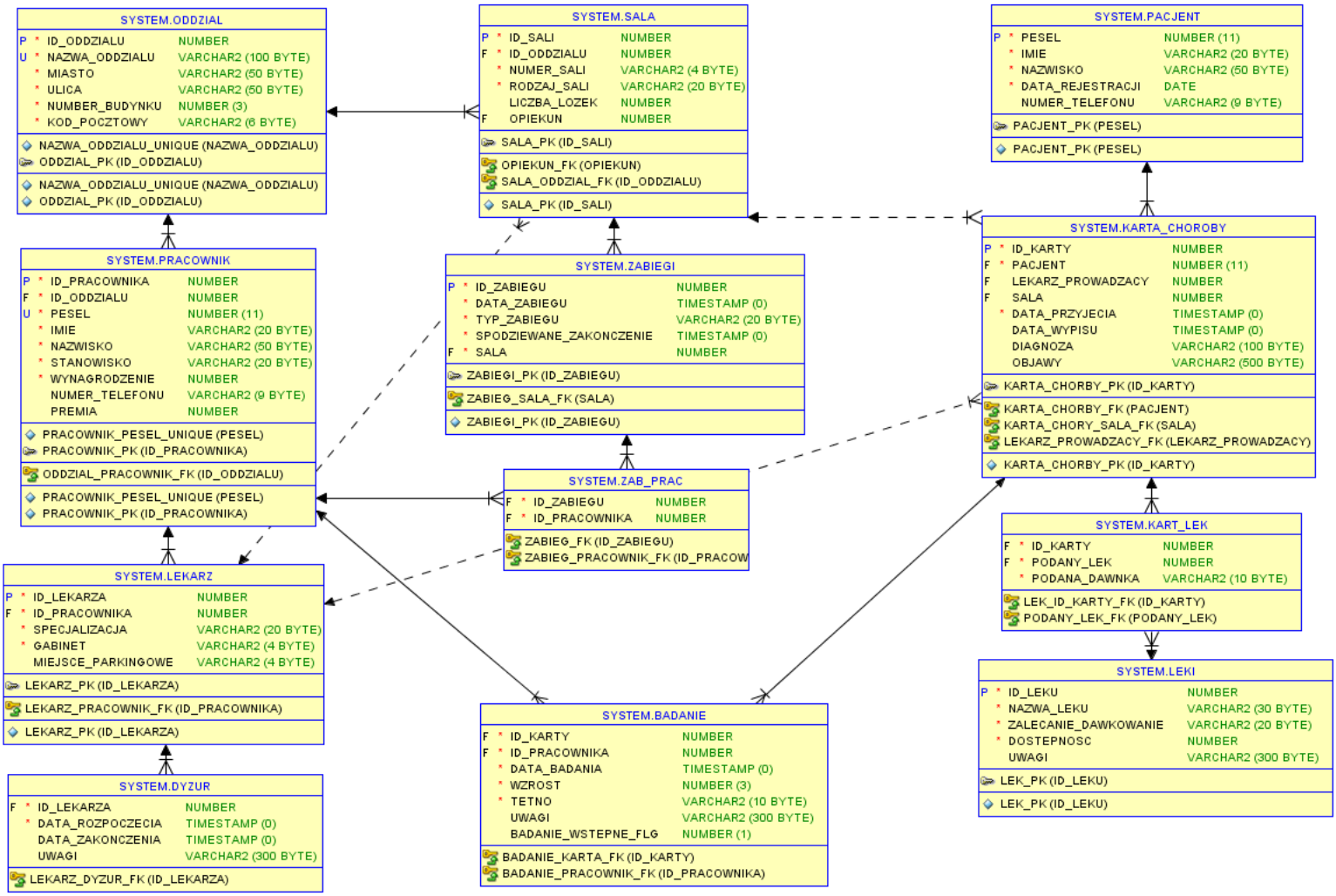
Bazy danych - mikroprojekt

Temat: Szpital	Grupa: 2ID14A	Karol Trociński Marek Szymański Dawid Szymkiewicz Kamil Świątek

1. Temat projektu

Tematem projektu było stworzenie bazy danych obsługującej szpital. Składa się ona z tabel: Oddział, pracownik, lekarz, dyżur, sala, pacjent, karta_choroby, badanie, leki, zabiegi oraz tabel realizujących związki wiele do wielu: kart_lek, zab_prac.

2. Diagram związku encji



3. Instrukcje Create

```
CREATE TABLE oddzial (
    id_oddzialu NUMBER CONSTRAINT oddzial_pk PRIMARY KEY,
    nazwa_oddzialu VARCHAR2(100) NOT NULL CONSTRAINT nazwa_oddzialu_unique UNIQUE,
    miasto VARCHAR2(50) NOT NULL,
    ulica VARCHAR2(50) NOT NULL,
    number_budynku NUMBER(3) NOT NULL,
    kod_pocztowy VARCHAR2(6) NOT NULL
);
```

```
CREATE TABLE pracownik (
    id_pracownika NUMBER CONSTRAINT pracownik_pk PRIMARY KEY,
    id_oddzialu NUMBER NOT NULL CONSTRAINT oddzial_pracownik_fk REFERENCES oddzial(id_oddzialu),
    PESEL NUMBER(11) NOT NULL CONSTRAINT pracownik_pesel_unique UNIQUE,
    imie VARCHAR2(20) NOT NULL,
    nazwisko VARCHAR2(50) NOT NULL,
    stanowisko VARCHAR2(20) NOT NULL,
    wynagrodzenie NUMBER NOT NULL CONSTRAINT wynagrodzenie_check CHECK(wynagrodzenie > 0 AND wynagrodzenie < 99999),
    numer_telefonu VARCHAR2(9),
    premia NUMBER DEFAULT 0
);
```

```
CREATE TABLE lekarz (
    id_lekarza NUMBER CONSTRAINT lekarz_pk PRIMARY KEY,
    id_pracownika NUMBER NOT NULL CONSTRAINT lekarz_pracownik_fk REFERENCES pracownik(id_pracownika),
    specjalizacja VARCHAR2(20) NOT NULL,
    gabinet VARCHAR2(4) NOT NULL,
    miejsce_parkingowe VARCHAR2(4)
);
```

```
CREATE TABLE dyzur (
    id_lekarza NUMBER NOT NULL CONSTRAINT lekarz_dyzur_fk REFERENCES lekarz(id_lekarza),
    data_rozpoczecia TIMESTAMP(0) DEFAULT SYSDATE NOT NULL,
    data_zakonczenia TIMESTAMP(0),
    uwagi VARCHAR2(300)
);
```

```
CREATE TABLE sala (
    id_sali NUMBER CONSTRAINT sala_pk PRIMARY KEY,
    id_oddzialu NUMBER NOT NULL CONSTRAINT sala_oddzial_fk REFERENCES oddzial(id_oddzialu),
    numer_sali VARCHAR2(4) NOT NULL, --LUB NUMBER
    rodzaj_sali VARCHAR2(20) NOT NULL,
    liczba_lozek NUMBER,
    opiekun NUMBER CONSTRAINT  opiekun_fk REFERENCES lekarz(id_lekarza)
);
```

```
CREATE TABLE pacjent (
    PESEL NUMBER(11) CONSTRAINT pacjent_pk PRIMARY KEY,
    imie VARCHAR2(20) NOT NULL,
    nazwisko VARCHAR2(50) NOT NULL,
    data_rejestracji DATE DEFAULT SYSDATE  NOT NULL,
    numer_telefonu VARCHAR2(9)
);
```

```sql
CREATE TABLE karta_choroby (
    id_karty NUMBER CONSTRAINT karta_chorby_pk PRIMARY KEY,
    pacjent NUMBER(11) NOT NULL CONSTRAINT karta_chorby_fk REFERENCES pacjent(PESEL),
    lekarz_prowadzacy NUMBER CONSTRAINT lekarz_prowadzacy_fk REFERENCES lekarz(id_lekarza),
    sala NUMBER CONSTRAINT karta_chory_sala_fk REFERENCES sala(id_sali),
    data_przyjecia TIMESTAMP(0) DEFAULT SYSDATE NOT NULL,
    data_wypisu TIMESTAMP(0) DEFAULT SYSDATE,
    diagnoza VARCHAR2(100),
    objawy VARCHAR2(500)
);

CREATE TABLE badanie (
    id_karty NOT NULL CONSTRAINT badanie_karta_fk REFERENCES karta_choroby(id_karty),
    id_pracownika NOT NULL CONSTRAINT badanie_pracownik_fk REFERENCES pracownik(id_pracownika),
    data_badania TIMESTAMP(0) DEFAULT SYSDATE NOT NULL,
    wzrost NUMBER(3) NOT NULL,
    tetno VARCHAR2(10) NOT NULL,
    uwagi VARCHAR2(300),
    badanie_wstepne_flg NUMBER(1) DEFAULT 0
);

CREATE TABLE leki (
    id_leku NUMBER CONSTRAINT lek_pk PRIMARY KEY,
    nazwa_leku VARCHAR2(30) NOT NULL,
    zalecanie_dawkowanie VARCHAR2(20) NOT NULL,
    dostepnosc NUMBER NOT NULL CONSTRAINT dostepnosc_check CHECK(dostepnosc >=0),
    uwagi VARCHAR2(300)
);

CREATE TABLE kart_lek (
    id_karty NUMBER NOT NULL CONSTRAINT lek_id_karty_fk REFERENCES karta_choroby(id_karty),
    podany_lek NUMBER NOT NULL CONSTRAINT podany_lek_fk REFERENCES leki(id_leku),
    podana_dawnka VARCHAR2(10) NOT NULL
);

CREATE TABLE zabiegi (
    id_zabiegu NUMBER CONSTRAINT zabiegi_pk PRIMARY KEY,
    data_zabiegu TIMESTAMP(0) DEFAULT SYSDATE NOT NULL,
    typ_zabiegu VARCHAR2(20) NOT NULL,
    spodziewane_zakonczenie TIMESTAMP(0) NOT NULL,
    sala NUMBER NOT NULL CONSTRAINT zabieg_sala_fk REFERENCES sala(id_sali)
);

CREATE TABLE zab_prac (
    id_zabiegu NUMBER NOT NULL CONSTRAINT zabieg_fk REFERENCES zabiegi(id_zabiegu),
    id_pracownika NUMBER NOT NULL CONSTRAINT zabieg_pracownik_fk REFERENCES pracownik(id_pracownika)
);
```

4. Widoki

```
CREATE OR REPLACE VIEW dyzury_w_okresie
AS
SELECT p.imie, p.nazwisko, p.pesel,d.data_rozpoczecia, d.data_zakonczenia FROM dyzur d
    JOIN Lekarz l ON l.id_lekarza = d.id_lekarza
    JOIN Pracownik p ON p.id_pracownika = l.id_pracownika;
```

Widok przechowuje informacje o dyżurach.

```
CREATE OR REPLACE VIEW Pracownicy_oddzialu
AS
SELECT p.imie, p.nazwisko, p.pesel, p.stanowisko, o.nazwa_oddzialu FROM Pracownik p
    JOIN oddzial o ON o.id_oddzialu = p.id_oddzialu;
```

Widok przechowuje informacje o pracownikach wraz z informacją o oddziale na którym pracują.

```
CREATE OR REPLACE VIEW zabiegi_dzisiaj
AS
SELECT p.imie, p.nazwisko, p.numer_telefonu, o.nazwa_oddzialu FROM Zabiegi z
    JOIN zab_prac zp ON z.id_zabiegu = zp.id_zabiegu
    JOIN pracownik p ON p.id_pracownika = zp.id_pracownika
    JOIN oddzial o ON o.id_oddzialu = p.id_oddzialu
    WHERE TO_CHAR(z.data_zabiegu, 'yyyy/mm/dd') = TO_CHAR(SYSDATE, 'yyyy/mm/dd')
    AND p.stanowisko = 'Lekarz';
```

Widok przechowuje informacje o zabiegach odbywających się „dzisiaj”.

```
CREATE OR REPLACE VIEW Zaplanowane_zabiegi
AS
SELECT z.data_zabiegu, z.typ_zabiegu, z.spodziewane_zakonczenie, o.nazwa_oddzialu, s.numer_sali FROM Zabiegi z
    JOIN sala s ON s.id_sali = z.sala
    JOIN oddzial o ON o.id_oddzialu = s.id_oddzialu
    WHERE z.data_zabiegu > SYSDATE;
```

Widok przechowuje informacje o zaplanowanych zabiegach, bez dzisiejszych.

```
CREATE OR REPLACE VIEW Podane_leki_przez_lekarza
AS
SELECT l.nazwa_leku, l.dostepnosc, l.uwagi, p.imie, p.nazwisko, p.pesel, lk.specjalizacja, lk.gabinet FROM Leki l
    JOIN kart_lek kl ON l.id_leku = kl.podany_lek
    JOIN karta_choroby kch ON kch.id_karty = kl.id_karty
    JOIN Lekarz lk ON lk.id_lekarza = kch.lekarz_prowadzacy
    JOIN Pracownik p ON p.id_pracownika = lk.id_pracownika;
```

Widok przechowuje informacje o lekach podanych przez danego lekarza.

```
CREATE OR REPLACE VIEW Pacjenci_na_miejscu
AS
SELECT p.imie, p.nazwisko, p.pesel, o.nazwa_oddzialu, s.numer_sali FROM Pacjent p
    JOIN karta_choroby kch ON kch.pacjent = p.pesel
    JOIN sala s ON s.id_sali = kch.sala
    JOIN oddzial o ON o.id_oddzialu = s.id_oddzialu
WHERE kch.data_wypisu IS NULL;
```

Widok przechowuje informacje o pacjentach wciąż przebywających w szpitalu.

5. Wyzwalacze

```
CREATE OR REPLACE TRIGGER przy_usuwaniu_pacjenta
BEFORE DELETE ON pacjent
FOR EACH ROW
DECLARE
BEGIN
    UPDATE karta_choroby SET karta_choroby.pacjent = 00000000000 WHERE karta_choroby.pacjent = :old.PESEL;
END;
/
```

Wyzwalacz anonimizuje kartę choroby przy wypisaniu pacjenta.

```
CREATE OR REPLACE TRIGGER przy_dodawaniu_pacjenta
AFTER INSERT ON pacjent
FOR EACH ROW
DECLARE
BEGIN
    INSERT INTO karta_choroby VALUES (karta_id_seq.NEXTVAL, :new.PESEL, NULL, NULL, SYSDATE, NULL, NULL, NULL);
END;
/
```

Wyzwalacz tworzy kartę choroby przy wstawieniu nowego pacjenta do tabeli pacjent

```
CREATE OR REPLACE TRIGGER przy_podaniu_leku
BEFORE INSERT ON KART_LEK
FOR EACH ROW
DECLARE
    aktualna_ilosc leki.dostepnosc%TYPE;
BEGIN
    SELECT l.dostepnosc INTO aktualna_ilosc FROM LEKI l WHERE l.id_leku = :new.podany_lek;
    IF aktualna_ilosc>0 THEN
        UPDATE LEKI SET DOSTEPNOSC =  DOSTEPNOSC - 1 WHERE id_leku = :new.podany_lek;
    END IF;
END;
/
```

Wyzwalacz aktualizuje dostępną ilość danego leku po podaniu go.

6. Kursory

Wypisywanie pacjentów ze stwierdzonym zgonem (Kursor)

```
CREATE OR REPLACE PROCEDURE wypisz_martwych
IS
    CURSOR martwi_pacjenci IS SELECT * FROM karta_choroby WHERE karta_choroby.diagnoza = 'Zgon' FOR UPDATE;
BEGIN
    FOR tmp IN martwi_pacjenci LOOP
        UPDATE karta_choroby SET data_wypisu = SYSDATE WHERE CURRENT OF martwi_pacjenci;
    END LOOP;
END;
/
```

Procedura ustawia datę wypisu pacjentom którzy został stwierdzony zgon.

Zamawianie leku (Kursor)

```
CREATE OR REPLACE FUNCTION oblicz_ilosc_podan(id_szuk_leku NUMBER)
RETURN NUMBER
IS
    ilosc_podawan NUMBER;
BEGIN
    SELECT count(*) INTO ilosc_podawan FROM kart_lek WHERE podany_lek = id_szuk_leku GROUP BY podany_lek;
    RETURN ilosc_podawan;
    EXCEPTION WHEN No_Data_Found THEN
        RETURN 0;
END;
/

CREATE OR REPLACE PROCEDURE leki_do_zamowienia
IS
    CURSOR lek_cur IS SELECT * FROM leki WHERE dostepnosc = 0;
    ilosc_podawan NUMBER;
BEGIN
    FOR tmp IN lek_cur LOOP
        ilosc_podawan := oblicz_ilosc_podan(tmp.id_leku);
        IF ilosc_podawan > 2 THEN
            UPDATE leki SET leki.uwagi = 'Zamówić, pline!' WHERE leki.id_leku = tmp.id_leku;
        ELSE
             UPDATE leki SET leki.uwagi = 'Zamówić' WHERE leki.id_leku = tmp.id_leku;
        END IF;
    END LOOP;

END;
/
```

Procedura pozwala na zamówienie leku.

Dodatek dyżurowy (Kursor)

```
CREATE OR REPLACE PROCEDURE dodatek_dyzurowy
IS
    CURSOR dyz_cur (id_szuk_lekarza NUMBER) IS
    SELECT EXTRACT(hour FROM (data_zakonczenia - data_rozpoczecia)) godziny, id_lekarza FROM dyzur
    WHERE dyzur.id_lekarza = id_szuk_lekarza AND
    EXTRACT(month from dyzur.data_zakonczenia) = EXTRACT(month FROM SYSDATE);
    CURSOR lek_cur IS SELECT * FROM lekarz l;
    liczba_godzin NUMBER := 0;
BEGIN
    FOR tmp_lek IN lek_cur LOOP
        liczba_godzin := 0;
        FOR tmp_dyz IN dyz_cur(tmp_lek.id_lekarza) LOOP
            liczba_godzin := liczba_godzin + tmp_dyz.godziny;
        END LOOP;
        IF liczba_godzin > 22 THEN
            UPDATE pracownik SET premia = 200 WHERE tmp_lek.id_pracownika = id_pracownika;
        END IF;
    END LOOP;
END;
/
```

Procedura dodaje dodatek lekarzom na podstawie godzin dyżurowych

7. Procedury i funkcje.

Dodawanie pacjenta

```
CREATE OR REPLACE PROCEDURE dodaj_pacjenta (
    PESEL_pacjenta pacjent.PESEL%TYPE,
    imie_pacjenta pacjent.imie%TYPE,
    nazwisko_pacjenta pacjent.nazwisko%TYPE,
    numer_telefonu_pacjenta pacjent.numer_telefonu%TYPE
    )
IS
    szukana_osoba pacjent%ROWTYPE;
BEGIN
    SELECT * INTO szukana_osoba FROM pacjent WHERE PESEL_pacjenta = pacjent.PESEL;
    dbms_output.ENABLE;
    dbms_output.put_line('Pacjent istnieje, zamiast tego przyjmij pacjenta');
    EXCEPTION WHEN No_Data_Found THEN
        INSERT INTO pacjent VALUES (PESEL_pacjenta, imie_pacjenta, nazwisko_pacjenta, sysdate, numer_telefonu_pacjenta);
END;
/
```

Procedura dodaje nowego pacjenta do bazy.

Przyjmowanie pacjenta do szpitala

```
CREATE OR REPLACE FUNCTION uzyskaj_id_oddzialu (nazwa_szukanego_oddzialu oddzial.nazwa_oddzialu%TYPE)
RETURN oddzial.id_oddzialu%TYPE
IS
    tmp oddzial%ROWTYPE;
BEGIN
    SELECT * INTO tmp FROM oddzial WHERE oddzial.nazwa_oddzialu = nazwa_szukanego_oddzialu;
    RETURN tmp.id_oddzialu;
    EXCEPTION WHEN No_Data_Found THEN
        RETURN 0;
END;
/
```

Funkcja zwraca id oddziału na który trafi pacjent.

```
CREATE OR REPLACE FUNCTION szukaj_sali (
    nazwa_szukanego_oddzialu oddzial.nazwa_oddzialu%TYPE,
    numer_szukanej_sali sala.numer_sali%TYPE
    )
RETURN sala.id_sali%TYPE
IS
    id_oddzialu_sali oddzial.id_oddzialu%TYPE;
    tmp sala%ROWTYPE;
    nie_ma_oddzialu EXCEPTION;
BEGIN
    id_oddzialu_sali := uzyskaj_id_oddzialu(nazwa_szukanego_oddzialu);
    IF id_oddzialu_sali != 0 THEN
        BEGIN
            SELECT * INTO tmp FROM sala
                WHERE sala.id_oddzialu = id_oddzialu_sali AND sala.numer_sali = numer_szukanej_sali;
        EXCEPTION when No_Data_Found THEN
            RETURN 0;
        END;
    ELSE
        RAISE nie_ma_oddzialu;
    END IF;
    RETURN tmp.id_sali;
    EXCEPTION WHEN nie_ma_oddzialu THEN
        RETURN -1;
END;
/
```

Funkcja zwraca id sali na którą trafi pacjent.

```
CREATE OR REPLACE FUNCTION dodaj_sale (
    PESEL_pacjenta pacjent.PESEL%TYPE,
    nazwa_oddzialu_pacjenta oddzial.nazwa_oddzialu%TYPE,
    numer_sali_pacjenta sala.numer_sali%TYPE
)
RETURN NUMBER
IS
    problem_sala EXCEPTION;
    id_szuk_sali NUMBER;
    tmp_pacjent pacjent%rowtype;
BEGIN
    SELECT * INTO tmp_pacjent FROM pacjent WHERE pacjent.pesel = PESEL_pacjenta;
    id_szuk_sali := szukaj_sali(nazwa_oddzialu_pacjenta, numer_sali_pacjenta);
    IF id_szuk_sali <= 0 THEN
        raise problem_sala;
    ELSE
        UPDATE karta_choroby SET sala = id_szuk_sali WHERE karta_choroby.pacjent = PESEL_pacjenta;
    END IF;

    return 0;

EXCEPTION WHEN No_data_found THEN
    return -3;
WHEN problem_sala THEN
    return id_szuk_sali - 1;
END;
/
```

Funkcja sprawdza czy nowy pacjent nie figuruje już w bazie (czy nie ma karty choroby)

```
CREATE OR REPLACE FUNCTION dodaj_sale (
    PESEL_pacjenta pacjent.PESEL%TYPE,
    nazwa_oddzialu_pacjenta oddzial.nazwa_oddzialu%TYPE,
    numer_sali_pacjenta sala.numer_sali%TYPE
)
RETURN NUMBER
IS
    problem_sala EXCEPTION;
    id_szuk_sali NUMBER;
    tmp_pacjent pacjent%rowtype;
BEGIN
    SELECT * INTO tmp_pacjent FROM pacjent WHERE pacjent.pesel = PESEL_pacjenta;
    id_szuk_sali := szukaj_sali(nazwa_oddzialu_pacjenta, numer_sali_pacjenta);
    IF id_szuk_sali <= 0 THEN
        raise problem_sala;
    ELSE
        UPDATE karta_choroby SET sala = id_szuk_sali WHERE karta_choroby.pacjent = PESEL_pacjenta;
    END IF;

    return 0;

EXCEPTION WHEN No_data_found THEN
    return -3;
WHEN problem_sala THEN
    return id_szuk_sali - 1;
END;
/
```

Funkcja dodaje id sali do pacjenta.

```
CREATE OR REPLACE PROCEDURE przyjmij_pacjenta (
    PESEL_pacjenta pacjent.PESEL%TYPE,
    nazwa_oddzialu_pacjenta oddzial.nazwa_oddzialu%TYPE,
    numer_sali_pacjenta sala.numer_sali%TYPE
)
IS
    tmp karta_choroby%ROWTYPE;
    tmp_pacjent pacjent%ROWTYPE;
    id_szukanej_sali NUMBER;
    pacjent_przyjety EXCEPTION;
    nie_ma_oddzalu EXCEPTION;
    nie_ma_sali EXCEPTION;
BEGIN
    SELECT * INTO tmp_pacjent FROM pacjent WHERE pacjent.PESEL = PESEL_pacjenta;

    IF sprawdz_karte_choroby(PESEL_pacjenta) THEN
        BEGIN
            id_szukanej_sali := szukaj_sali( nazwa_oddzialu_pacjenta, numer_sali_pacjenta);
            IF id_szukanej_sali = 0 THEN
                RAISE nie_ma_sali;
            ELSIF id_szukanej_sali = -1 THEN
                RAISE nie_ma_oddzalu;
            ELSE
                INSERT INTO karta_choroby VALUES (karta_id_seq.NEXTVAL, PESEL_pacjenta, NULL, id_szukanej_sali, sysdate, NULL, NULL, NULL);
            END IF;

            EXCEPTION WHEN nie_ma_sali THEN
                dbms_output.ENABLE;
                dbms_output.put_line('Nie ma takiej sali');
            WHEN nie_ma_oddzalu THEN
                dbms_output.ENABLE;
                dbms_output.put_line('Nie ma takiego oddzalu!');
        END;
    ELSE
        RAISE pacjent_przyjety;
    END IF;

    EXCEPTION WHEN pacjent_przyjety THEN
        dbms_output.ENABLE;
        dbms_output.put_line('Taki pacient zostal przyjety!');
    WHEN No_Data_Found THEN
        dbms_output.ENABLE;
        dbms_output.put_line('Taki pacient nie istnieje!');
END;
```

Procedura przyjmuje pacjenta (Tworzy mu kartę choroby i dodaje mu rekordy takie jak id sali i oddziału na którym się znajduje)
WYJĄTKI gdy: podano złą sale, podano zły oddział, pacjent był wcześniej przyjęty, taki pacjent nie został zarejestrowany.

Wypisz pacjenta

```
CREATE OR REPLACE PROCEDURE wypisz_pacjenta(PESEL_pacjenta karta_choroby.pacjent%TYPE)
IS BEGIN
    IF sprawdz_karte_choroby(PESEL_pacjenta) THEN
        dbms_output.ENABLE;
        dbms_output.put_line('Pacjent nie istnieje lub zosta³ ju¿ wypisany');
    ELSE
        UPDATE karta_choroby SET karta_choroby.data_wypisu = sysdate WHERE karta_choroby.pacjent = PESEL_pacjenta;
    END IF;
END;
/
```

Procedura ustawia datę wypisania w karcie choroby.
WYJĄTKI gdy: pacjent został już wypisany.

Dodawanie diagnozy

```
CREATE OR REPLACE PROCEDURE dodaj_diagnoze(
    PESEL_pacjenta karta_choroby.pacjent%TYPE,
    diagnoza_dla_pacjenta karta_choroby.diagnoza%TYPE
    )
IS BEGIN
    IF sprawdz_karte_choroby(PESEL_pacjenta) THEN
        dbms_output.ENABLE;
        dbms_output.put_line('Pacjent nie istnieje lub zosta³ ju¿ wypisany');
    ELSE
        UPDATE karta_choroby SET karta_choroby.diagnoza = diagnoza_dla_pacjenta WHERE karta_choroby.pacjent = PESEL_pacjenta;
    END IF;
END;
/
```

Procedura dodaje diagnozę do karty choroby.
WYJĄTKI gdy: pacjent nie istnieje lub został już wypisany.

Dodawanie objawów

```
CREATE OR REPLACE PROCEDURE dodaj_objawy(
    PESEL_pacjenta karta_choroby.pacjent%TYPE,
    objawy_pacjenta karta_choroby.objawy%TYPE
    )
IS BEGIN
    IF sprawdz_karte_choroby(PESEL_pacjenta) THEN
        dbms_output.ENABLE;
        dbms_output.put_line('Pacjent nie istnieje lub zosta³ ju¿ wypisany');
    ELSE
        UPDATE karta_choroby SET karta_choroby.objawy = objawy_pacjenta WHERE karta_choroby.pacjent = PESEL_pacjenta;
    END IF;
END;
/
```

Procedura dodaje objawy do karty choroby.
WYJĄTKI gdy: pacjent nie istnieje lub został już wypisany.

Dodawanie badania

```
CREATE OR REPLACE FUNCTION znajdz_pracownika(
    PESEL_pracownika pracownik.PESEL%TYPE
    )
    RETURN BOOLEAN
    IS
        tmp pracownik.id_pracownika%TYPE := 0;
    BEGIN
        SELECT p.PESEL INTO tmp FROM pracownik p WHERE p.PESEL = PESEL_pracownika;
        IF tmp = 0 THEN
            RETURN FALSE;
        ELSE
            RETURN TRUE;
        END IF;
        EXCEPTION WHEN no_data_found THEN
            RETURN FALSE;

END;
/
```

Funkcja zwraca „true” jeżeli znaleziono podanego pracownika, w przeciwnym razie „false”;

```
CREATE OR REPLACE PROCEDURE dodaj_badanie(
    data_badania badanie.data_badania%TYPE,
    PESEL_pacjenta karta_choroby.pacjent%TYPE,
    wzrost_pacjenta badanie.wzrost%TYPE,
    tetno_pacjenta badanie.tetno%TYPE,
    uwagi_pacjenta badanie.uwagi%TYPE,
    badanie_wstepne_pacjenta badanie.badanie_wstepne_flg%TYPE,
    PESEL_pracownika pracownik.PESEL%TYPE
)
IS
    id_karty_pacjenta karta_choroby.id_karty%TYPE := 0;
    czy_pracownik_istnieje BOOLEAN;
    karta_istnieje BOOLEAN;
    id_znalezionego_pracownika NUMBER;
    nie_ma_karty EXCEPTION;
    nie_ma_pracownika EXCEPTION;
BEGIN
   czy_pracownik_istnieje := znajdz_pracownika(PESEL_pracownika);
    IF czy_pracownik_istnieje = FALSE THEN
        RAISE nie_ma_pracownika;
    ELSE
        BEGIN
            karta_istnieje := sprawdz_karte_choroby(PESEL_pacjenta);
            IF karta_istnieje = TRUE THEN
                RAISE nie_ma_karty;
            ELSE
                SELECT id_karty INTO id_karty_pacjenta FROM karta_choroby WHERE karta_choroby.pacjent = PESEL_pacjenta AND karta_choroby.data_wypisu IS NULL;
                SELECT id_pracownika INTO id_znalezionego_pracownika FROM pracownik WHERE pracownik.PESEL = PESEL_pracownika;
                INSERT INTO badanie VALUES(id_karty_pacjenta, id_znalezionego_pracownika, data_badania, wzrost_pacjenta, tetno_pacjenta, uwagi_pacjenta, badanie_wstepne_pacjenta);
            END IF;
            EXCEPTION WHEN nie_ma_karty THEN
                dbms_output.ENABLE;
                dbms_output.put_line('Nie ma takiego pacjenta');
            WHEN no_data_found THEN
               dbms_output.ENABLE;
                dbms_output.put_line('Nie ma takiego pacjenta');
        END;
    END IF;
    EXCEPTION WHEN nie_ma_pracownika THEN
        dbms_output.ENABLE;
        dbms_output.put_line('Nie ma takiego pracownika');
END;
/
```

Procedura dodaje badanie dla pacjenta o podanym numerze PESEL WYJĄTKI gdy: Nie znaleziono pacjenta o takim numerze PESEL, nie znaleziono podanego pracownika który miał przeprowadzać badanie.

Dodawanie zabiegu

```
CREATE OR REPLACE FUNCTION znajdz_sale(
    p_numer_sali sala.numer_sali%TYPE
)
RETURN NUMBER
IS
    tmp sala.id_sali%TYPE;
BEGIN
    BEGIN
        SELECT id_sali INTO tmp FROM sala WHERE p_numer_sali=sala.numer_sali;
        EXCEPTION WHEN NO_DATA_FOUND THEN
            tmp:=NULL;
    END;
    RETURN tmp;
END;
/
```

Funkcja zwraca id sali o podanym numerze.

```
CREATE OR REPLACE FUNCTION znajdz_rodzaj_sali(
    p_id_sali sala.id_sali%TYPE
)
RETURN sala.rodzaj_sali%TYPE
IS
    tmp sala.rodzaj_sali%TYPE;
BEGIN
    SELECT rodzaj_sali INTO tmp FROM sala WHERE id_sali=p_id_sali;
    RETURN tmp;
END;
/
```

Funkcja zwraca rodzaj sali o podanym id.

```
CREATE OR REPLACE FUNCTION znajdz_najwieksze_id_zabiegu
RETURN zabiegi.id_zabiegu%TYPE
IS
    tmp zabiegi.id_zabiegu%TYPE;
BEGIN
    SELECT MAX(id_zabiegu) INTO tmp FROM zabiegi;
    IF tmp IS NULL THEN
        tmp:=0;
    END IF;
    RETURN tmp;
END;
/
```

Funkcja zwraca największe id zabiegu.

```
CREATE OR REPLACE FUNCTION znajdz_id_pracownika(
    p_pesel pracownik.PESEL%TYPE
)
RETURN pracownik.id_pracownika%TYPE
IS
    tmp pracownik.id_pracownika%TYPE;
BEGIN
    SELECT id_pracownika INTO tmp FROM pracownik WHERE PESEL=p_pesel;
    IF tmp IS NULL THEN
        tmp:=0;
    END IF;
    RETURN tmp;
END;
/
```

Funkcja zwraca id pracownika o podanym numerze PESEL

```
CREATE OR REPLACE FUNCTION znajdz_stanowisko(
    p_id_pracownika pracownik.id_pracownika%TYPE
)
RETURN pracownik.stanowisko%TYPE
IS
    tmp pracownik.stanowisko%TYPE;
BEGIN
    SELECT stanowisko INTO tmp FROM pracownik WHERE id_pracownika=p_id_pracownika;
    RETURN tmp;
END;
/
```

Funkcja zwraca stanowisko pracownika o podanym id.

```
CREATE OR REPLACE PROCEDURE dodaj_zabieg(
    p_data_zabiegu zabiegi.data_zabiegu%TYPE,
    p_typ_zabiegu zabiegi.typ_zabiegu%TYPE,
    p_spodziewane_zakonczenie zabiegi.spodziewane_zakonczenie%TYPE,
    p_numer_sali sala.numer_sali%TYPE,
    p_pesel_lekarza pracownik.PESEL%TYPE
)
IS
    tmp_data_poczatku zabiegi.data_zabiegu%TYPE;
    tmp_koniec zabiegi.spodziewane_zakonczenie%TYPE;
    p_id_sali zabiegi.sala%TYPE;
    p_id_zabiegu zabiegi.id_zabiegu%TYPE;
    p_id_prowadzacego pracownik.id_pracownika%TYPE;
    cursor cur_data_poczatku is select data_zabiegu from zabiegi where p_id_sali=sala;
    cursor cur_koniec is select spodziewane_zakonczenie from zabiegi where p_id_sali=sala;
    sala_zajeta EXCEPTION;
    nie_ma_sali EXCEPTION;
    sala_nie_zabiegowa EXCEPTION;
    czas EXCEPTION;
    nie_lekarz EXCEPTION;
BEGIN
    p_id_zabiegu:=znajdz_najwieksze_id_zabiegu();
    p_id_zabiegu:=p_id_zabiegu+1;
    p_id_sali:=znajdz_sale(p_numer_sali);
    p_id_prowadzacego:=znajdz_id_pracownika(p_pesel_lekarza);
    IF p_id_sali IS NULL THEN
        RAISE nie_ma_sali;
    ELSIF p_data_zabiegu>p_spodziewane_zakonczenie THEN
        RAISE czas;
    ELSIF znajdz_rodzaj_sali(p_id_sali) NOT LIKE 'Zabiegowe' AND znajdz_rodzaj_sali(p_id_sali) NOT LIKE 'Operacyjne' THEN
        RAISE sala_nie_zabiegowa;
    ELSIF p_id_prowadzacego IS NULL THEN
        RAISE nie_lekarz;
    ELSIF znajdz_stanowisko(p_id_prowadzacego) NOT LIKE 'Lekarz' THEN
        RAISE nie_lekarz;
    END IF;
    OPEN cur_data_poczatku;
    OPEN cur_koniec;
    LOOP
    FETCH cur_data_poczatku INTO tmp_data_poczatku;
    FETCH cur_koniec INTO tmp_koniec;
    EXIT WHEN cur_data_poczatku%NOTFOUND;
    EXIT WHEN cur_koniec%NOTFOUND;

    IF (tmp_data_poczatku IS NOT NULL AND tmp_koniec IS NOT NULL) AND ((p_data_zabiegu>=tmp_data_poczatku AND p_spodziewane_zakonczenie<=tmp_koniec)) THEN
        RAISE sala_zajeta;
    END IF;
    END LOOP;
    INSERT INTO zabiegi VALUES(p_id_zabiegu, p_data_zabiegu, p_typ_zabiegu, p_spodziewane_zakonczenie, p_id_sali);
    INSERT INTO zab_prac VALUES(p_id_zabiegu, p_id_prowadzacego);
    EXCEPTION WHEN sala_zajeta THEN
        dbms_output.ENABLE;
        dbms_output.put_line('Ta sala jest juz zarezerwowana!');
    WHEN nie_ma_sali THEN
        dbms_output.ENABLE;
        dbms_output.put_line('Taka sala nie istnieje');
    WHEN sala_nie_zabiegowa THEN
        dbms_output.ENABLE;
        dbms_output.put_line('Wybrana sala nie jest zabiegowa!');
    WHEN czas THEN
        dbms_output.ENABLE;
        dbms_output.put_line('Data konca zabiegu nie moze byc mniejsza nic data poczatku');
    WHEN nie_lekarz THEN
        dbms_output.ENABLE;
        dbms_output.put_line('Prowadzacy zabieg musi istniec i byc lekarzem');
END;
/
```

Procedura dodaje nowy zabieg. Użyte zostały dwa kursory, jeden wczytuje daty początku zabiegów, drugi – końca.

WYJĄTKI gdy: sala w której ma odbyć się zabieg jest o podanej porze zajęta, podana sala nie istnieje, podana sala nie jest zabiegowa lub operacyjna, data końca nie zgadza się z datą początku, prowadzący zabieg nie jest lekarzem.

8. Instrukcje Insert

Odziały

```
-- Oddziały

INSERT INTO oddzial VALUES (oddzial_id_seq.NEXTVAL, 'Pediatria', 'Kielce', 'Grunwaldzka', '1', '25-736');
INSERT INTO oddzial VALUES (oddzial_id_seq.NEXTVAL, 'Chirurgia', 'Kielce', 'Grunwaldzka', '2', '25-736');
INSERT INTO oddzial VALUES (oddzial_id_seq.NEXTVAL, 'Okulistyka', 'Kielce', 'Grunwaldzka', '3', '25-736');
INSERT INTO oddzial VALUES (oddzial_id_seq.NEXTVAL, 'Kardiologia', 'Kielce', 'Grunwaldzka', '4', '25-736');
INSERT INTO oddzial VALUES (oddzial_id_seq.NEXTVAL, 'Neurologia', 'Kielce', 'Grunwaldzka', '5', '25-736');
INSERT INTO oddzial VALUES (oddzial_id_seq.NEXTVAL, 'Laryngologia', 'Kielce', 'Grunwaldzka', '6', '25-736');
INSERT INTO oddzial VALUES (oddzial_id_seq.NEXTVAL, 'Ortopedia', 'Kielce', 'Grunwaldzka', '7', '25-736');
INSERT INTO oddzial VALUES (oddzial_id_seq.NEXTVAL, 'Neonatologiczny', 'Kielce', 'Grunwaldzka', '8', '25-736');
INSERT INTO oddzial VALUES (oddzial_id_seq.NEXTVAL, 'Nefrologia', 'Kielce', 'Grunwaldzka', '9', '25-736');
INSERT INTO oddzial VALUES (oddzial_id_seq.NEXTVAL, 'Psychiatria', 'Kielce', 'Grunwaldzka', '10', '25-736');
INSERT INTO oddzial VALUES (oddzial_id_seq.NEXTVAL, 'Paliacja', 'Kielce', 'Grunwaldzka', '11', '25-736');
```

Pracownicy

```
-- Pracownicy

INSERT INTO pracownik VALUES (pracownik_id_seq.NEXTVAL, 1, 64111492484, 'Marian', 'Konewka', 'Konserwator', 2500, '111222333', 0);
INSERT INTO pracownik VALUES (pracownik_id_seq.NEXTVAL, 1, 82020353229, 'Marianna', 'Konewka', 'Sprz瑿aczka', 2500, '111222334', 0);
INSERT INTO pracownik VALUES (pracownik_id_seq.NEXTVAL, 2, 82061365746, 'Zdzis歟w', 'Nowak', 'Konserwator', 2600, '543222333', 0);
INSERT INTO pracownik VALUES (pracownik_id_seq.NEXTVAL, 2, 69100625638,  'Kornel', 'Tutaj', 'Piel瑿niarz', 2800, '111753333', 0);
INSERT INTO pracownik VALUES (pracownik_id_seq.NEXTVAL, 3, 72082757257, 'Joanna', 'Papaj', 'Piel瑿niarka', 2800, '111222412', 0);
INSERT INTO pracownik VALUES (pracownik_id_seq.NEXTVAL, 4, 79012333534, 'Kamil', 'Wakszmucki', 'Piel瑿niarz', 2800, '111983333', 0);
INSERT INTO pracownik VALUES (pracownik_id_seq.NEXTVAL, 10, 88062245348, 'Andrzej', 'Kolejarz', 'Ochroniarz', 2300, '532222333', 0);
INSERT INTO pracownik VALUES (pracownik_id_seq.NEXTVAL, 10, 74051626245, 'Wac歟w', 'Tam', 'Ochroniarz', 2300, '111981333', 0);
INSERT INTO pracownik VALUES (pracownik_id_seq.NEXTVAL, 5, 63060275674, 'Dawid', 'Szynka', 'Informatyk', 9999, '111324333', 0);
INSERT INTO pracownik VALUES (pracownik_id_seq.NEXTVAL, 6, 74092039181, 'Karolina', 'Strata', 'Sekretarka', 2900, '991222333', 0);
INSERT INTO pracownik VALUES (pracownik_id_seq.NEXTVAL, 3, 84100454185, 'Wasilij', 'Papataj', 'Piel瑿niarz', 2800, '854222412', 0);
INSERT INTO pracownik VALUES (pracownik_id_seq.NEXTVAL, 4, 70100973119, 'Antonina', 'Michalczyk', 'Piel瑿niarka', 2800, '643983333', 0);
INSERT INTO pracownik VALUES (pracownik_id_seq.NEXTVAL, 1, 90103091696, 'Dawid', 'Szynka', 'Ochroniarz', 2300, '532532333', 0);
INSERT INTO pracownik VALUES (pracownik_id_seq.NEXTVAL, 10, 89061881285, 'Tomasz', 'Problem', 'Bazodanowiec', 9999, '765981976', 0);
INSERT INTO pracownik VALUES (pracownik_id_seq.NEXTVAL, 3, 64082272753, 'Karol', 'Marcinkowski', 'Technik', 3000, '331676333', 0);
INSERT INTO pracownik VALUES (pracownik_id_seq.NEXTVAL, 7, 61091781393, 'Magda', 'Pionel', 'Sekretarka', 2900, '991222999', 0);


INSERT INTO pracownik VALUES (pracownik_id_seq.NEXTVAL, 1, 84070986833, 'Wies歟w', 'M瑿zka', 'Lekarz', 6800, '987983333', 0);
INSERT INTO pracownik VALUES (pracownik_id_seq.NEXTVAL, 2, 79123175858, 'Andrzej', 'Martyniuk', 'Lekarz', 6800, '532222333', 0);
INSERT INTO pracownik VALUES (pracownik_id_seq.NEXTVAL, 3, 80111927216, 'Karmil', 'Czekaj', 'Lekarz', 6800, '987981333', 0);
INSERT INTO pracownik VALUES (pracownik_id_seq.NEXTVAL, 4, 61042382833, 'Teresa', 'Witam', 'Lekarz', 6800, '111324999', 0);
INSERT INTO pracownik VALUES (pracownik_id_seq.NEXTVAL, 5, 70043085676, 'Micha□', 'Rze□ik', 'Lekarz', 6800, '991786333', 0);
INSERT INTO pracownik VALUES (pracownik_id_seq.NEXTVAL, 6, 72092738482, 'Pawe□', 'Tombreno', 'Lekarz', 6800, '854563412', 0);
INSERT INTO pracownik VALUES (pracownik_id_seq.NEXTVAL, 7, 84030337567, 'Paulina', 'Stonoga', 'Lekarz', 6800, '643983757', 0);
INSERT INTO pracownik VALUES (pracownik_id_seq.NEXTVAL, 8, 69062869271, 'Joanna', 'Kaszanka', 'Lekarz', 6800, '532532999', 0);
INSERT INTO pracownik VALUES (pracownik_id_seq.NEXTVAL, 9, 82102172779, 'Diego', 'Manino', 'Lekarz', 6800, '974981976', 0);
INSERT INTO pracownik VALUES (pracownik_id_seq.NEXTVAL, 10, 76011152665, 'Marian', 'Pazera', 'Lekarz', 6800, '331676395', 0);
INSERT INTO pracownik VALUES (pracownik_id_seq.NEXTVAL, 11, 76112152365, 'Dorbromi歟', 'Ogier', 'Lekarz', 6800, '991864999', 0);
INSERT INTO pracownik VALUES (pracownik_id_seq.NEXTVAL, 5, 12334512376, 'Lena', 'Zielinska', 'Lekarz', 6500, '567103871', 0);
--lekarze


INSERT INTO lekarz VALUES (lekarz_id_seq.NEXTVAL, 17, 'Pediatria', '150c', '142b');
INSERT INTO lekarz VALUES (lekarz_id_seq.NEXTVAL, 18, 'Chirurgia', '150c', '142b');
INSERT INTO lekarz VALUES (lekarz_id_seq.NEXTVAL, 19, 'Okulistyka', '150c', '142b');
INSERT INTO lekarz VALUES (lekarz_id_seq.NEXTVAL, 20, 'Kardiologia', '150c', '142b');
INSERT INTO lekarz VALUES (lekarz_id_seq.NEXTVAL, 21, 'Neurologia', '150c', '142b');
INSERT INTO lekarz VALUES (lekarz_id_seq.NEXTVAL, 22, 'Laryngologia', '150c', '142b');
INSERT INTO lekarz VALUES (lekarz_id_seq.NEXTVAL, 23, 'Ortopedia', '150c', '142b');
INSERT INTO lekarz VALUES (lekarz_id_seq.NEXTVAL, 24, 'Neonatologiczny', '150c', '142b');
INSERT INTO lekarz VALUES (lekarz_id_seq.NEXTVAL, 25, 'Nefrologia', '150c', '142b');
INSERT INTO lekarz VALUES (lekarz_id_seq.NEXTVAL, 26, 'Psychiatria', '150c', '142b');
INSERT INTO lekarz VALUES (lekarz_id_seq.NEXTVAL, 27, 'Paliacja', '150c', '142b');
INSERT INTO lekarz VALUES (lekarz_id_seq.NEXTVAL, 28, 'Neurologia', '111c', '45');
INSERT INTO lekarz VALUES (lekarz_id_seq.NEXTVAL, 29, 'Laryngologia', '118a', '37');
INSERT INTO lekarz VALUES (lekarz_id_seq.NEXTVAL, 30, 'Ortopedia', '301', '46');
INSERT INTO lekarz VALUES (lekarz_id_seq.NEXTVAL, 31, 'Kardiologia', '291', '91');
```

Lekarze
Sale

```
--sale
INSERT INTO sala VALUES (sala_id_seq.NEXTVAL, 1, '1a', 'Zabiegowe', 10, 1);
INSERT INTO sala VALUES (sala_id_seq.NEXTVAL, 2, '2a', 'Zabiegowe', 10, 2);
INSERT INTO sala VALUES (sala_id_seq.NEXTVAL, 3, '3a', 'Zabiegowe', 10, 3);
INSERT INTO sala VALUES (sala_id_seq.NEXTVAL, 3, '3b', 'Operacyjne', 10, 3);
INSERT INTO sala VALUES (sala_id_seq.NEXTVAL, 3, '3c', 'Pooperacyjne', 10, 3);
INSERT INTO sala VALUES (sala_id_seq.NEXTVAL, 4, '4a', 'Zabiegowe', 10, 4);
INSERT INTO sala VALUES (sala_id_seq.NEXTVAL, 5, '5a', 'Zabiegowe', 10, 5);
INSERT INTO sala VALUES (sala_id_seq.NEXTVAL, 6, '6a', 'Zabiegowe', 10, 6);
INSERT INTO sala VALUES (sala_id_seq.NEXTVAL, 6, '6b', 'Operacyjne', 10, 6);
INSERT INTO sala VALUES (sala_id_seq.NEXTVAL, 6, '6c', 'Pooperacyjne', 10, 6);
INSERT INTO sala VALUES (sala_id_seq.NEXTVAL, 7, '7a', 'Zabiegowe', 10, 7);
INSERT INTO sala VALUES (sala_id_seq.NEXTVAL, 8, '8a', 'Zabiegowe', 10, 8);
INSERT INTO sala VALUES (sala_id_seq.NEXTVAL, 9, '9a', 'Zabiegowe', 10, 9);
INSERT INTO sala VALUES (sala_id_seq.NEXTVAL, 10, '10a', 'Izolatka', 1, 10);
INSERT INTO sala VALUES (sala_id_seq.NEXTVAL, 10, '10b', 'Izolatka', 1, 10);
INSERT INTO sala VALUES (sala_id_seq.NEXTVAL, 10, '10c', 'Izolatka', 1, 10);
INSERT INTO sala VALUES (sala_id_seq.NEXTVAL, 10, '10d', 'Izolatka', 1, 10);
INSERT INTO sala VALUES (sala_id_seq.NEXTVAL, 10, '10e', 'Izolatka', 1, 10);
INSERT INTO sala VALUES (sala_id_seq.NEXTVAL, 11, '11a', 'Zabiegowe', 10, 11);
```

Pacjenci

```
--Pacjenci
INSERT INTO Pacjent VALUES (00000000000, 'Pacjent', 'Anonimowy', TO_DATE('1970/01/01', 'yyyy/mm/dd'), '000000000');
INSERT INTO Pacjent VALUES (87030836913, 'Andrzej', 'Komoda', TO_DATE('2019/01/01', 'yyyy/mm/dd'), '547043400');
INSERT INTO Pacjent VALUES (67061098799, 'Kacper', 'Ko這dziejczyk', TO_DATE('2018/12/19', 'yyyy/mm/dd'), '467086537');
INSERT INTO Pacjent VALUES (83060349216, 'Antoni', 'Czerwiec', TO_DATE('2017/01/12', 'yyyy/mm/dd'), '782579803');
INSERT INTO Pacjent VALUES (90082265516, 'Leszek', 'Kropydlak', TO_DATE('2016/12/23', 'yyyy/mm/dd'), '263214944');
INSERT INTO Pacjent VALUES (69100469296, 'Kilian', 'Fallens', TO_DATE('2018/11/09', 'yyyy/mm/dd'), '863214945');
INSERT INTO Pacjent VALUES (72101999152, 'Ryszard', 'Skanu', TO_DATE('2018/08/02', 'yyyy/mm/dd'), '569523585');
INSERT INTO Pacjent VALUES (84010867194, 'Arkadiusz', 'Szybczak', TO_DATE('2018/07/09', 'yyyy/mm/dd'), '569443585');
INSERT INTO Pacjent VALUES (87061092623, 'Hanna', 'Mostowiak', TO_DATE('2017/07/12', 'yyyy/mm/dd'), '367330735');
INSERT INTO Pacjent VALUES (99122599541, 'Anna', 'Jurczak', TO_DATE('2019/01/08', 'yyyy/mm/dd'), '553221648');
INSERT INTO Pacjent VALUES (54112977426, 'Janina', 'Kostecka', TO_DATE('2018/11/12', 'yyyy/mm/dd'), '67756728');
INSERT INTO Pacjent VALUES (73082781622, 'Bogus敬wa', 'Linda', TO_DATE('2018/02/14', 'yyyy/mm/dd'), '866685659');
INSERT INTO Pacjent VALUES (73102672925, 'Rozalia', 'Cyrkon', TO_DATE('2019/01/02', 'yyyy/mm/dd'), '382371863');
INSERT INTO Pacjent VALUES (53070577686, 'Iwona', 'Kajzerka', TO_DATE('2017/01/13', 'yyyy/mm/dd'), '995126457');
INSERT INTO Pacjent VALUES (90061259387, 'Anna', 'Fatygowa', TO_DATE('2018/12/12', 'yyyy/mm/dd'), '266663539');
INSERT INTO Pacjent VALUES (85062869592, 'Kacper','Korde址a', TO_DATE('2016/09/11', 'yyyy/mm/dd'), '352856740');
INSERT INTO Pacjent VALUES (51101212652, 'Magda', 'Wressler', TO_DATE('2016/09/11', 'yyyy/mm/dd'), '777771216');
```

Karty choroby (fragment)

```
--Karta choroby
INSERT INTO karta_choroby VALUES (karta_id_seq.NEXTVAL, 51101212652, 15, 6, TO_DATE('2016/09/11 17-41', 'yyyy/mm/dd HH24-MI'), TO_DATE('2016/09/11
INSERT INTO karta_choroby VALUES (karta_id_seq.NEXTVAL, 90082265516,  9, 13, TO_DATE('2016/12/23 11-29', 'yyyy/mm/dd HH24-MI'), TO_DATE('2017/01/0
INSERT INTO karta_choroby VALUES (karta_id_seq.NEXTVAL, 83060349216,  11, 6, TO_DATE('2017/01/12 22-41', 'yyyy/mm/dd HH24-MI'), TO_DATE('2017/01/1
INSERT INTO karta_choroby VALUES (karta_id_seq.NEXTVAL, 87061092623, 10, 14, TO_DATE('2017/07/12 14-14', 'yyyy/mm/dd HH24-MI'), TO_DATE('2017/07/1
INSERT INTO karta_choroby VALUES (karta_id_seq.NEXTVAL, 87061092623, 11, 19, TO_DATE('2017/07/22 13-34', 'yyyy/mm/dd HH24-MI'), TO_DATE('2017/07/2
INSERT INTO karta_choroby VALUES (karta_id_seq.NEXTVAL, 73082781622, 14, 1, TO_DATE('2018/02/14 11-10', 'yyyy/mm/dd HH24-MI'), TO_DATE('2018/02/15
INSERT INTO karta_choroby VALUES (karta_id_seq.NEXTVAL, 84010867194, 7, 11, TO_DATE('2018/07/09 13-13', 'yyyy/mm/dd HH24-MI'), TO_DATE('2018/07/10
INSERT INTO karta_choroby VALUES (karta_id_seq.NEXTVAL, 72101999152,  6, 8, TO_DATE('2018/08/02 03-59', 'yyyy/mm/dd HH24-MI'), TO_DATE('2018/08/14
INSERT INTO karta_choroby VALUES (karta_id_seq.NEXTVAL, 69100469296,  6, 10, TO_DATE('2018/11/09 07-59', 'yyyy/mm/dd HH24-MI'), TO_DATE('2018/12/2
INSERT INTO karta_choroby VALUES (karta_id_seq.NEXTVAL, 87030836913,  4, 6, TO_DATE('2019/01/01 12-41', 'yyyy/mm/dd HH24-MI'), NULL, NULL, 'Mocny
INSERT INTO karta_choroby VALUES (karta_id_seq.NEXTVAL, 83060349216,  11, 6, TO_DATE('2019/01/01 20-33', 'yyyy/mm/dd HH24-MI'), TO_DATE('2019/01/0
INSERT INTO karta_choroby VALUES (karta_id_seq.NEXTVAL, 73102672925, 3, 5, TO_DATE('2019/01/02 21-10', 'yyyy/mm/dd HH24-MI'), NULL, NULL, 'Silny b
INSERT INTO karta_choroby VALUES (karta_id_seq.NEXTVAL, 99122599541, 2, 2, TO_DATE('2019/01/08 18-20', 'yyyy/mm/dd HH24-MI'), NULL, NULL, 'Silny b
INSERT INTO karta_choroby VALUES (karta_id_seq.NEXTVAL, 67061098799,  5, 7, TO_DATE('2018/12/19 07-21', 'yyyy/mm/dd HH24-MI'), NULL, NULL, 'Gor敓z
INSERT INTO karta_choroby VALUES (karta_id_seq.NEXTVAL, 51101212652, 15, 6, TO_DATE('2019/01/16 11-00', 'yyyy/mm/dd HH24-MI'), NULL, NULL, 'Duszno
```

Leki

```
--Leki

INSERT INTO leki VALUES(lek_id_seq.NEXTVAL, 'Adrenalina WZF', '300 ?/0,3 ml', 200, NULL);
INSERT INTO leki VALUES(lek_id_seq.NEXTVAL, 'Antytoksyna botulinowa ABE', '5000 j.m', 10, NULL);
INSERT INTO leki VALUES(lek_id_seq.NEXTVAL, 'Asentra', '100 mg', 15, NULL);
INSERT INTO leki VALUES(lek_id_seq.NEXTVAL, 'Benodil', '0,125 mg/ml', 0, NULL);
INSERT INTO leki VALUES(lek_id_seq.NEXTVAL, 'Bepanthen Eye', ' ', 5, 'Jedno opakowanie blisko ko?ca daty przydatn?i');
INSERT INTO leki VALUES(lek_id_seq.NEXTVAL, 'Biseptol', '240 mg/5 ml', 23, NULL);
INSERT INTO leki VALUES(lek_id_seq.NEXTVAL, 'Bunondol (tabletki)', '0,2 mg', 30, NULL); --Przeciwb?lowy
INSERT INTO leki VALUES(lek_id_seq.NEXTVAL, 'Bunondol (zastrzyki)', '0,2 mg', 30, NULL); --Przeciwb?lowy
INSERT INTO leki VALUES(lek_id_seq.NEXTVAL, 'Galvus', '50 mg', 0, NULL);
INSERT INTO leki VALUES(lek_id_seq.NEXTVAL, 'GalvuGeloplasma', '100 ml', 3, NULL);
INSERT INTO leki VALUES(lek_id_seq.NEXTVAL, 'Desloratadine Genoptim', ' 0,5 mg/ml', 7, NULL);
INSERT INTO leki VALUES(lek_id_seq.NEXTVAL, 'Detreman', '1 mg', 8, NULL);
INSERT INTO leki VALUES(lek_id_seq.NEXTVAL, 'Lercan', '10 mg', 2, NULL);
INSERT INTO leki VALUES(lek_id_seq.NEXTVAL, 'Raevretan', '3mg', 7, NULL);
INSERT INTO leki VALUES(lek_id_seq.NEXTVAL, 'Opelol', '20 ml/2mg', 13, NULL);
```

Leki-Pacjenci

```
--Lek-Pacj
INSERT INTO kart_lek VALUES(4, 3, '100mg');
INSERT INTO kart_lek VALUES(4, 3, '200mg');
INSERT INTO kart_lek VALUES(2, 6, '240mg');
INSERT INTO kart_lek VALUES(5, 1, '300 ?');
INSERT INTO kart_lek VALUES(8, 7, '0,1 mg');
INSERT INTO kart_lek VALUES(8, 1, '300 ?');
INSERT INTO kart_lek VALUES(3, 15, '1 mg');
INSERT INTO kart_lek VALUES(14, 7, '0,1 mg');
INSERT INTO kart_lek VALUES(5, 14, '3mg');
INSERT INTO kart_lek VALUES(1, 7, '0.2mg');
INSERT INTO kart_lek VALUES(11, 2, '0.5 mg');
INSERT INTO kart_lek VALUES(4, 5, '1 mg');
INSERT INTO kart_lek VALUES(6, 8, '0.2 mg');
INSERT INTO kart_lek VALUES(7, 10, '100 ml');
INSERT INTO kart_lek VALUES(13, 6, '240 mg');
```

Dyżury

```
-- Dyzury

INSERT INTO dyzur VALUES(1, TO_DATE('2019/01/11 17-00', 'yyyy/mm/dd HH24-MI'), TO_DATE('2019/01/12 07-00', 'yyyy/mm/dd HH24-MI'), NULL);
INSERT INTO dyzur VALUES(1, TO_DATE('2019/01/13 08-00', 'yyyy/mm/dd HH24-MI'), TO_DATE('2019/01/13 19-00', 'yyyy/mm/dd HH24-MI'), NULL);
INSERT INTO dyzur VALUES(1, TO_DATE('2019/01/15 08-00', 'yyyy/mm/dd HH24-MI'), TO_DATE('2019/01/15 19-00', 'yyyy/mm/dd HH24-MI'), NULL);
INSERT INTO dyzur VALUES(2, TO_DATE('2019/01/11 08-00', 'yyyy/mm/dd HH24-MI'), TO_DATE('2019/01/11 19-00', 'yyyy/mm/dd HH24-MI'), NULL);
INSERT INTO dyzur VALUES(2, TO_DATE('2019/01/13 17-00', 'yyyy/mm/dd HH24-MI'), TO_DATE('2019/01/14 07-00', 'yyyy/mm/dd HH24-MI'), NULL);
INSERT INTO dyzur VALUES(3, TO_DATE('2019/01/16 08-00', 'yyyy/mm/dd HH24-MI'), TO_DATE('2019/01/16 19-00', 'yyyy/mm/dd HH24-MI'), NULL);

INSERT INTO dyzur VALUES(4, TO_DATE('2018/12/11 17-00', 'yyyy/mm/dd HH24-MI'), TO_DATE('2018/12/12 07-00', 'yyyy/mm/dd HH24-MI'), NULL);
INSERT INTO dyzur VALUES(4, TO_DATE('2018/12/13 08-00', 'yyyy/mm/dd HH24-MI'), TO_DATE('2018/12/13 19-00', 'yyyy/mm/dd HH24-MI'), NULL);
INSERT INTO dyzur VALUES(4, TO_DATE('2018/12/15 08-00', 'yyyy/mm/dd HH24-MI'), TO_DATE('2018/12/15 19-00', 'yyyy/mm/dd HH24-MI'), NULL);
INSERT INTO dyzur VALUES('7', TO_DATE('2019/01/07 08-00', 'yyyy/mm/dd HH24-MI'), TO_DATE('2019/01/07 15-00', 'yyyy/mm/dd HH24-MI'), NULL);
INSERT INTO dyzur VALUES('8', TO_DATE('2019/01/07 15-00', 'yyyy/mm/dd HH24-MI'), TO_DATE('2019/01/07 23-00', 'yyyy/mm/dd HH24-MI'), NULL);
INSERT INTO dyzur VALUES('1', TO_DATE('2019/01/07 23-00', 'yyyy/mm/dd HH24-MI'), TO_DATE('2019/01/08 07-00', 'yyyy/mm/dd HH24-MI'), NULL);
INSERT INTO dyzur VALUES('15', TO_DATE('2019/01/07 08-00', 'yyyy/mm/dd HH24-MI'), TO_DATE('2019/01/07 15-00', 'yyyy/mm/dd HH24-MI'), NULL);
INSERT INTO dyzur VALUES('3', TO_DATE('2019/01/07 15-00', 'yyyy/mm/dd HH24-MI'), TO_DATE('2019/01/07 23-00', 'yyyy/mm/dd HH24-MI'), NULL);
INSERT INTO dyzur VALUES('11', TO_DATE('2019/01/07 23-00', 'yyyy/mm/dd HH24-MI'), TO_DATE('2019/01/08 07-00', 'yyyy/mm/dd HH24-MI'), NULL);
```

Badania

```
-- Badania

INSERT INTO badanie VALUES(1, 17, TO_DATE('2016/01/11 18-00', 'yyyy/mm/dd HH24-MI'), 173, '70/min', 'brak', 0);
INSERT INTO badanie VALUES(2, 17, TO_DATE('2017/01/03 08-10', 'yyyy/mm/dd HH24-MI'), 173, '60/min', 'brak', 0);
INSERT INTO badanie VALUES(3, 19, TO_DATE('2017/01/14 12-20', 'yyyy/mm/dd HH24-MI'), 173, '70/min', 'brak', 0);
INSERT INTO badanie VALUES(4, 19, TO_DATE('2017/01/12 16-00', 'yyyy/mm/dd HH24-MI'), 173, '60/min', 'brak', 0);
INSERT INTO badanie VALUES(5, 20, TO_DATE('2017/01/22 17-00', 'yyyy/mm/dd HH24-MI'), 173, '85/min', 'brak', 0);
INSERT INTO badanie VALUES(6, 21, TO_DATE('2018/01/15 11-00', 'yyyy/mm/dd HH24-MI'), 173, '70/min', 'brak', 0);
INSERT INTO badanie VALUES(7, 22, TO_DATE('2018/01/10 13-00', 'yyyy/mm/dd HH24-MI'), 173, '85/min', 'brak', 1);
INSERT INTO badanie VALUES(8, 19, TO_DATE('2018/01/14 09-00', 'yyyy/mm/dd HH24-MI'), 173, '70/min', 'brak', 0);
INSERT INTO badanie VALUES(9, 24, TO_DATE('2018/01/24 09-00', 'yyyy/mm/dd HH24-MI'), 173, '60/min', 'brak', 0);
INSERT INTO badanie VALUES(10, 26, TO_DATE('2018/01/14 09-00', 'yyyy/mm/dd HH24-MI'), 173, '70/min', 'brak', 0);
INSERT INTO badanie VALUES(11, 17, TO_DATE('2018/01/03 12-00', 'yyyy/mm/dd HH24-MI'), 173, '85/min', 'brak', 1);
INSERT INTO badanie VALUES(12, 20, TO_DATE('2019/01/26 15-00', 'yyyy/mm/dd HH24-MI'), 173, '70/min', 'brak', 1);
INSERT INTO badanie VALUES(13, 19, TO_DATE('2019/01/25 20-05', 'yyyy/mm/dd HH24-MI'), 173, '70/min', 'brak', 0);
INSERT INTO badanie VALUES(14, 30, TO_DATE('2019/01/30 13-10', 'yyyy/mm/dd HH24-MI'), 173, '60/min', 'brak', 0);
INSERT INTO badanie VALUES(15, 29, TO_DATE('2019/01/16 14-00', 'yyyy/mm/dd HH24-MI'), 173, '85/min', 'brak', 0);
```

Zabiegi

```
-- Zabiegi

INSERT INTO zabiegi VALUES(1, TO_DATE('2016/01/12 17-00', 'yyyy/mm/dd HH24-MI'), 'Zastrzyk', TO_DATE('2016/01/12 18-20', 'yyyy/mm/dd HH24-MI'), 1);
INSERT INTO zab_prac VALUES(1, 17);
INSERT INTO zabiegi VALUES(2, TO_DATE('2016/01/20 13-00', 'yyyy/mm/dd HH24-MI'), 'Zastrzyk', TO_DATE('2016/01/20 14-20', 'yyyy/mm/dd HH24-MI'), 2);
INSERT INTO zab_prac VALUES(2, 17);
INSERT INTO zabiegi VALUES(3, TO_DATE('2016/01/31 10-00', 'yyyy/mm/dd HH24-MI'), 'Opatrzenie', TO_DATE('2016/01/31 10-34', 'yyyy/mm/dd HH24-MI'), 2);
INSERT INTO zab_prac VALUES(3, 19);
INSERT INTO zabiegi VALUES(4, TO_DATE('2016/01/27 11-00', 'yyyy/mm/dd HH24-MI'), 'Operacja', TO_DATE('2016/01/27 16-00', 'yyyy/mm/dd HH24-MI'), 4);
INSERT INTO zab_prac VALUES(4, 20);
INSERT INTO zab_prac VALUES(4, 17);
INSERT INTO zab_prac VALUES(4, 4);
INSERT INTO zabiegi VALUES(5, TO_DATE('2016/01/23 12-00', 'yyyy/mm/dd HH24-MI'), 'Szczepienie', TO_DATE('2016/01/23 12-20', 'yyyy/mm/dd HH24-MI'), 11);
INSERT INTO zab_prac VALUES(5, 5);
INSERT INTO zabiegi VALUES(6, TO_DATE('2016/01/17 14-00', 'yyyy/mm/dd HH24-MI'), 'Zastrzyk', TO_DATE('2016/01/17 15-00', 'yyyy/mm/dd HH24-MI'), 12);
INSERT INTO zab_prac VALUES(6, 30);
INSERT INTO zabiegi VALUES(7, TO_DATE('2016/01/12 17-00', 'yyyy/mm/dd HH24-MI'), 'Zastrzyk', TO_DATE('2016/01/12 17-10', 'yyyy/mm/dd HH24-MI'), 13);
INSERT INTO zab_prac VALUES(7, 4);
INSERT INTO zabiegi VALUES(8, TO_DATE('2016/01/03 19-00', 'yyyy/mm/dd HH24-MI'), 'Gips', TO_DATE('2016/01/03 19-30', 'yyyy/mm/dd HH24-MI'), 1);
INSERT INTO zab_prac VALUES(8, 17);
INSERT INTO zabiegi VALUES(9, TO_DATE('2016/01/09 18-05', 'yyyy/mm/dd HH24-MI'), 'Zastrzyk', TO_DATE('2016/01/09 18-15', 'yyyy/mm/dd HH24-MI'), 12);
INSERT INTO zab_prac VALUES(9, 4);
INSERT INTO zabiegi VALUES(10, TO_DATE('2016/01/13 13-15', 'yyyy/mm/dd HH24-MI'), 'Szczepienie', TO_DATE('2016/01/13 13-30', 'yyyy/mm/dd HH24-MI'), 1);
INSERT INTO zab_prac VALUES(10, 4);
```

9. Funkcjonalność klienta

Na potrzeby projektu powstał klient graficzny, oferujący podaną niżej funkcjonalność:

- obsługa widoków z parametrami
- wykonywanie własnych zapytań do bazy danych
- dodawanie pacjenta
- przyjmowanie pacjenta
- wypisywanie pacjenta
- dodanie diagnozy
- dodanie objawów
- dodanie badania
- dodanie zabiegu
- zamawianie leków
- wypisywanie pacjentów u których stwierdzono zgon
- dodanie dodatku