Procedury

```
#Procedura odpowiedzialna za aktualizowanie danych w tabeli wizyty i odpowiednie usuwanie
#danych z tabeli pracownicy i użytkownicy  po zwolnieniu pracownika ==  usunięciu z bazy

# transakcja do zwalniania pracowników
DROP PROCEDURE deleteUsers;
CALL deleteUsers('');

DELIMITER $$
CREATE PROCEDURE deleteUsers(IN name varchar(30))
  BEGIN
    START TRANSACTION ;
    SELECT id_użytkownika FROM użytkownicy WHERE Login in(name) into @a; #
kwestie bezpieczeństwa dlatego in;
    SET FOREIGN_KEY_CHECKS = 0;
    UPDATE wizyty SET status_wizyty = 'do przełożenia', lekarz = '0' where
lekarz = @a;
    SET FOREIGN_KEY_CHECKS = 1;
    DELETE FROM dyzury WHERE id_lekarz = @a;
    DELETE FROM uslugi where id_wykonawca = @a;
    DELETE from lekarz where id_lekarz = @a;
    DELETE FROM urlopy where pracownik = @a;
    DELETE FROM użytkownicy where id_użytkownika = @a;
    DELETE FROM pracownicy where id_pracownik = @a;
    COMMIT;
  END $$
DELIMITER ;

DELIMITER $$
CREATE PROCEDURE deleteEmployee(IN id int)
 BEGIN
   START TRANSACTION ;
   SET FOREIGN_KEY_CHECKS = 0;
   UPDATE wizyty SET status_wizyty = 'do przełożenia', lekarz = '0' where lekarz = id;
   SET FOREIGN_KEY_CHECKS = 1;
   DELETE FROM dyzury WHERE id_lekarz = id;
   DELETE FROM uslugi where id_wykonawca = id;
   DELETE from lekarz where id_lekarz = id;
   DELETE FROM urlopy where pracownik = id;
   DELETE FROM użytkownicy where id_użytkownika = id;
   DELETE FROM pracownicy where id_pracownik = id;
   COMMIT;
 END $$
DELIMITER ;




#Procedura która zmienia status wszystkich wizyt, które się już odbyły na
"zakończona".
call statusofVisit();
DROP PROCEDURE statusofVisit;
DELIMITER $$
```

```
CREATE PROCEDURE statusofVisit()
BEGIN
  START TRANSACTION;
  UPDATE wizyty SET status_wizyty = 'zakończona' WHERE data_wizyty <
CURDATE();
  COMMIT ;
END $$
DELIMITER ;




#Procedura, która dodaje losową ilość pacjentów podawaną jako argument
„ile”. Posłużyła ona do stworzenia 1 mln rekordów w tabeli pacjenci. Można
zmieniac dla roznorodnosci danych imiona i nazwiska itp. w tabeli
tymczasowej dane_pomocnicze

DELIMITER $$
CREATE PROCEDURE DodajPacjenta(in ile int)
BEGIN
  DECLARE rok CHAR(4);
  DECLARE miesiac CHAR(2);
  DECLARE dzien CHAR(2);
  DECLARE rokpeselu CHAR(2);
  DECLARE miesiacpeselu CHAR(2);
  DECLARE dzienpeselu char(2);
  DECLARE koniecpeselu INT;
  DECLARE i INT;
  DECLARE PESELpoczatek CHAR(10);
  DECLARE ostatnia_cyfra CHAR(1);
  DECLARE suma INT;
  DECLARE suma_string CHAR(5);
  SET i = 1;

  CREATE TEMPORARY TABLE if not exists dane_pomocnicze
  (
    imie     varchar(20),
    nazwisko varchar(20),
    ulica    varchar(20),
    miasto   varchar(20)
  );
  INSERT INTO dane_pomocnicze VALUES ('Zbigniew', 'Zieliński',
'Nowowiejska', 'Sopot');
  INSERT INTO dane_pomocnicze VALUES ('Ryszard', 'Lewandoski', 'Prusa',
'Wrocław');
  INSERT INTO dane_pomocnicze VALUES ('Zygfryd', 'Wawrzyniak',
'Wyszyńskiego', 'Katowice');
  INSERT INTO dane_pomocnicze VALUES ('Natalia', 'Gałka', 'Grunwaldzka',
'Legnica');
  INSERT INTO dane_pomocnicze VALUES ('Przemysław', 'Tytoń', 'Słowiańska',
'Działoszyn');
  INSERT INTO dane_pomocnicze VALUES ('Jolanta', 'Wieczorek',
'Nadodrzańska', 'Wieruszów');
  INSERT INTO dane_pomocnicze VALUES ('Mikołaj', 'Kopernik', 'Bogata',
'Piotrków Trybunalski');
  INSERT INTO dane_pomocnicze VALUES ('Monika', 'Lisek', 'Przestarzala',
'Gdańsk');
  INSERT INTO dane_pomocnicze VALUES ('Katarzyna', 'Łabądź', 'Pobliska',
'Mirków');
  INSERT INTO dane_pomocnicze VALUES ('Patrycja', 'Łakowska', 'Złota',
'Częstochowa');
```

```sql
  INSERT INTO dane_pomocnicze VALUES ('Radek', 'Rembecki', 'Narodowa',
'Wieruszów');
  INSERT INTO dane_pomocnicze VALUES ('Mateusz', 'Ciołek', 'Kleczkowska',
'Odessa');
  INSERT INTO dane_pomocnicze VALUES ('Arkadiusz', 'Wrona', 'Trzebnicka',
'Bolesławiec');
  INSERT INTO dane_pomocnicze VALUES ('Łukasz', 'Szczęsny', 'Zmyślona',
'Stara Wieś');
  INSERT INTO dane_pomocnicze VALUES ('Albert', 'Wacłowski', 'Szkolna',
'Nowa Wieś');
  INSERT INTO dane_pomocnicze VALUES ('Agnieszka', 'Nogaj', 'Wieluńska',
'Zaolzie');
  INSERT INTO dane_pomocnicze VALUES ('Paulina', 'Figiel', 'Zakładowa',
'Wejcherowo');
  INSERT INTO dane_pomocnicze VALUES ('Jowita', 'Dzięcioł', 'Kwiatowa',
'Tyble');
  INSERT INTO dane_pomocnicze VALUES ('Bogdan', 'Tyc', 'Brązowa',
'Bralin');
  INSERT INTO dane_pomocnicze VALUES ('Adam', 'Wkręta', 'Energetycka',
'Oleśnica');
  INSERT INTO dane_pomocnicze VALUES ('Piotr', 'Zawisza', 'Przesiecka',
'Zduńska Wola');
  INSERT INTO dane_pomocnicze VALUES ('Włodzimiesz', 'Szafrański',
'Mickiewicza', 'Piotrków Trybunalski');
  INSERT INTO dane_pomocnicze VALUES ('Bartłomiej', 'Kumaty', 'Marii
Konopnickiej', 'Radom');
  INSERT INTO dane_pomocnicze VALUES ('Dominik', 'Gasztych', 'Legionów',
'Warszawa');
  INSERT INTO dane_pomocnicze VALUES ('Aneta', 'Sobera', '3 Maja',
'Szczecin');
  INSERT INTO dane_pomocnicze VALUES ('Agata', 'Lias', 'Racławicka', 'Rabka
Zdrój');
  INSERT INTO dane_pomocnicze VALUES ('Adrian', 'Stankiewicz', 'Krakowska',
'Oleśnica');
  INSERT INTO dane_pomocnicze VALUES ('Alicja', 'Mordor', 'Stara', 'Wizły
Małe');
  INSERT INTO dane_pomocnicze VALUES ('Eustachy', 'Mura', 'Długa',
'Zakopane');
  INSERT INTO dane_pomocnicze VALUES ('Zenona', 'Ludowicz', 'Nowa',
'Opole');

  loop_data: WHILE i <= ile DO
  SET rok = (SELECT FLOOR((RAND() * (100)) + 1911));
  SET miesiac = (SELECT FLOOR((RAND() * (12)) + 1));
  SET dzien = (SELECT FLOOR((RAND() * (28)) + 1));
  SET rokpeselu = SUBSTRING(rok, 3, 2);
  SET miesiacpeselu = miesiac;

  # dodajemy +20 do miesiąca w peselu dla urodzonych po roku 1999
  IF rok >= 2000 THEN
    SET miesiacpeselu = miesiacpeselu + 20;
  END IF;

  SET dzienpeselu = dzien;
  SET koniecpeselu = (SELECT FLOOR((RAND() * (9000)) + 1000));


  IF CHAR_LENGTH(rokpeselu) = 1 THEN
    SET rokpeselu = CONCAT(0, rokpeselu);
  END IF;
  IF CHAR_LENGTH(miesiacpeselu) = 1 THEN
```

```
    SET miesiacpeselu = CONCAT(0, miesiacpeselu);
  END IF;
  IF CHAR_LENGTH(miesiac) = 1
  THEN
    SET miesiac = CONCAT(0, miesiac);
  END IF;
  IF CHAR_LENGTH(dzienpeselu) = 1 THEN
    SET dzienpeselu = CONCAT(0, dzienpeselu);
  END IF;


  SET PESELpoczatek = CONCAT(rokpeselu, miesiacpeselu, dzienpeselu,
koniecpeselu);
  SET suma = (9 * CAST(SUBSTRING(PESELpoczatek, 1, 1) AS UNSIGNED)
    + 7 * CAST(SUBSTRING(PESELpoczatek, 2, 1) AS UNSIGNED)
    + 3 * CAST(SUBSTRING(PESELpoczatek, 3, 1) AS UNSIGNED)
    + 1 * CAST(SUBSTRING(PESELpoczatek, 4, 1) AS UNSIGNED)
    + 9 * CAST(SUBSTRING(PESELpoczatek, 5, 1) AS UNSIGNED)
    + 7 * CAST(SUBSTRING(PESELpoczatek, 6, 1) AS UNSIGNED)
    + 3 * CAST(SUBSTRING(PESELpoczatek, 7, 1) AS UNSIGNED)
    + 1 * CAST(SUBSTRING(PESELpoczatek, 8, 1) AS UNSIGNED)
    + 9 * CAST(SUBSTRING(PESELpoczatek, 9, 1) AS UNSIGNED)
    + 7 * CAST(SUBSTRING(PESELpoczatek, 10, 1) AS UNSIGNED));

  SET suma_string = CONCAT(suma, 'X');
  SET ostatnia_cyfra = SUBSTRING(suma_string, CHAR_LENGTH(suma_string) - 1,
1);
  SET @PESEL = CONCAT(PESELpoczatek, ostatnia_cyfra);

  IF @PESEL IN (Select pesel from pacjent) THEN
    ITERATE loop_data;
  end if ;

  SET @imie = (SELECT imie FROM dane_pomocnicze ORDER BY RAND() LIMIT 1);
  SET @nazwisko = (SELECT nazwisko FROM dane_pomocnicze ORDER BY RAND()
LIMIT 1);
  SET @data_urodzenia = CONCAT(rok, '-', miesiac, '-', dzienpeselu);
  SET @nr_telefonu = (SELECT FLOOR((RAND() * (99999999)) + 500000000));
  SET @ulica = (SELECT ulica FROM dane_pomocnicze ORDER BY RAND() LIMIT 1);
  SET @miasto = (SELECT miasto FROM dane_pomocnicze ORDER BY RAND() LIMIT
1);

  SET @adres = CONCAT(@miasto, ', ', 'ul. ', @ulica, ' ', FLOOR((RAND() *
(99)) + 1));

  INSERT INTO pacjent (Imie, Nazwisko, Pesel, nr_telefonu, Data_urodzenia,
Adres)
  VALUES (@imie, @nazwisko, @PESEL, @nr_telefonu, @data_urodzenia, @adres);
  SET i = i + 1;
  END WHILE loop_data;
  DROP table dane_pomocnicze;
END $$ DELIMITER ;

drop procedure DodajPacjenta;
CALL DodajPacjenta(100000);
```

```
# Procedura do sprawdzania poprawności uprawnień przy logowaniu

  DELIMITER $$
DROP PROCEDURE IF EXISTS checkPrivileges $$
CREATE PROCEDURE checkPrivileges(IN log VARCHAR(50), IN has varchar(50))
BEGIN
  SET @query = CONCAT("SELECT Stanowsko FROM użytkownicy WHERE Login = '", log, "' AND
Hasło = '", has, "' ;");
  PREPARE stmt FROM @query;
  EXECUTE stmt;
  DEALLOCATE PREPARE stmt;
END $$
DELIMITER;

CALL checkPrivileges('lek@2', '5fg6kczrgc');
```

```
# Procedura do dodawania konkretnego pacjenta o konretnych danych

DELIMITER $$
DROP PROCEDURE IF EXISTS addPatient$$
CREATE PROCEDURE addPatient(IN Imie VARCHAR(30),
                            IN Nazwisko VARCHAR(30),
                            IN Pesel CHAR(11),
                            IN nr_telefonu int(9),
                            IN Data_urodzenia DATE,
                            IN Adres VARCHAR(150))
BEGIN
  SET @result = CONCAT("INSERT INTO pacjent(Imie, Nazwisko, Pesel, nr_telefonu,
Data_urodzenia, Adres)
   VALUES ('", Imie, "','", Nazwisko, "','", Pesel, "','", nr_telefonu, "','", Data_urodzenia, "','", Adres,
"');");
  PREPARE stmt FROM @result;
  EXECUTE stmt;
  DEALLOCATE PREPARE stmt;
END $$
DELIMITER ;
CALL addPatient('Anna', 'Drabik', '84071369929', '567845992', '1984-07-13', 'Wrocław,
ul.Grunwaldzka 32');
```

```
# Procedura do dodawania konkretnego pracownika o podanych danych

DELIMITER $$
DROP PROCEDURE IF EXISTS addWorker$$
CREATE PROCEDURE addWorker(IN Imie VARCHAR(30),
                           IN Nazwisko VARCHAR(30),
                           IN Data_urodzenia DATE,
                           IN nr_telefonu int(9),
                           IN Pesel CHAR(11),
                           IN Pensja int(11),
                           IN Adres VARCHAR(150))
```

```sql
BEGIN
  SET @result = CONCAT("INSERT INTO pracownicy(Imie, Nazwisko, 
Data_urodzenia, nr_telefonu, Pesel, Pensja, Adres)
   VALUES ('", Imie, "','", Nazwisko, "','", Data_urodzenia, "','", 
nr_telefonu, "','", Pesel, "','", Pensja, "','", 
                       Adres, "');");
  PREPARE stmt FROM @result;
  EXECUTE stmt;
  DEALLOCATE PREPARE stmt;
END $$
DELIMITER ;
CALL addWorker('Anna', 'Drabik', '1984-07-13', '567845993', '84071369929', 
'5000', 'Wrocław, ul.Grunwaldzka 32');






#Procedura do dodawania nowych konkretnych wizyt
DELIMITER $$
DROP PROCEDURE IF EXISTS addVisit$$
CREATE PROCEDURE addVisit(IN lekarz INT(11),
                          IN pacjent INT(11),
                          IN gabinet INT(11),
                          IN data_wizyty DATE,
                          IN czas_wizyty ENUM ('30','90','60', '120'),
                          IN id_usługi INT(11),
                          IN priotytet ENUM ('1','2','3'),
                          IN status_wizyty ENUM ('zakończona', 
'oczekująca', 'do przełożenia'))
BEGIN
  SET @result = CONCAT("INSERT INTO wizyty(lekarz, pacjent, gabinet, 
data_wizyty, czas_wizyty, id_usługi, priorytet, status_wizyty)
   VALUES ('", lekarz, "','", pacjent, "','", gabinet, "','", data_wizyty, 
"','", czas_wizyty, "','", id_usługi, "','", 
                       priotytet, "','", status_wizyty, "');");
  PREPARE stmt FROM @result;
  EXECUTE stmt;
  DEALLOCATE PREPARE stmt;
END $$
DELIMITER ;
CALL addVisit('5', '4', '10', '2019-05-08', '60', '2', '1', 'oczekująca');






# Procedura do dodawania nowych zabiegów

DELIMITER $$
DROP PROCEDURE IF EXISTS addZabieg$$
CREATE PROCEDURE addZabieg(IN nazwa VARCHAR(30))
BEGIN
  SET @result = CONCAT("INSERT INTO zabieg(nazwa)
   VALUES ('", nazwa, "');");
  PREPARE stmt FROM @result;
  EXECUTE stmt;
  DEALLOCATE PREPARE stmt;
END $$
```

```sql
DELIMITER ;
CALL addZabieg('Polerowanie');




# Procedura do dodawania nowych usług

DELIMITER $$
DROP PROCEDURE IF EXISTS addUsluga;
CREATE PROCEDURE addUsluga(IN id_wykonawca INT(11),
                           IN cena INT(11),
                           IN zabieg_id INT(11))
BEGIN
  SET @result = CONCAT("INSERT INTO uslugi(id_wykonawca, cena, zabieg_id)
   VALUES ('", id_wykonawca, "','", cena, "','", zabieg_id, "');");
  PREPARE stmt FROM @result;
  EXECUTE stmt;
  DEALLOCATE PREPARE stmt;
END $$
DELIMITER ;
CALL addUsluga(5, 100, 15);




# Procedura do dodawania nowych dyżurów

DELIMITER $$
DROP PROCEDURE IF EXISTS addDyzur;
CREATE PROCEDURE addDyzur(IN id_lekarz INT(11),
                          IN dzien varchar(30),
                          IN poczatek time(2),
                          IN koniec time(2))
BEGIN
  SET @result = CONCAT("INSERT INTO dyzury(id_lekarz, dzien, poczatek,
koniec)
   VALUES ('", id_lekarz, "','", dzien, "','", poczatek, "','", koniec,
"');");
  PREPARE stmt FROM @result;
  EXECUTE stmt;
  DEALLOCATE PREPARE stmt;
END $$
DELIMITER ;
CALL addDyzur(4, 'Tuesday', '16:00', '18:00');




#procedura do dodawania urlopów

DELIMITER $$
DROP PROCEDURE IF EXISTS addUrlop;
CREATE PROCEDURE addUrlop(IN pracownik INT(11),
                    IN początek_urlopu DATE,
                    IN koniec_urlopu DATE)
BEGIN
  SET @result = CONCAT("INSERT INTO urlopy(pracownik, początek_urlopu, koniec_urlopu)
   VALUES ('", pracownik, "','", początek_urlopu, "','", koniec_urlopu, "');");
  PREPARE stmt FROM @result;
  EXECUTE stmt;
  DEALLOCATE PREPARE stmt;
END $$
DELIMITER ;
CALL addUrlop(10, '2019-05-25', '2018-05-15');




#procedura do dodawania określonej ilości wizyt

DELIMITER $$
DROP PROCEDURE IF EXISTS createVisit;
CREATE PROCEDURE createVisit(IN number int)
BEGIN
  SET @counter = 1;
#   SELECT DATE_SUB(CURDATE(), INTERVAL 365 DAY) INTO @date;
  SELECT DATE_ADD(curdate(),INTERVAL 1 day) INTO @date;
  SET @doctor = (SELECT MIN(id_lekarz) from lekarz);
  SET @count_doctor = (SELECT count(*) from lekarz);
  SET @min_doctor = (SELECT MIN(id_lekarz) from lekarz);
  SET @count_patient = (SELECT count(*) from pacjent);

  WHILE @counter <= number DO

    SELECT poczatek from dyzury where id_lekarz = @doctor AND dzien =
DAYNAME(@date) INTO @start;
    SELECT koniec from dyzury where id_lekarz = @doctor AND dzien =
DAYNAME(@date) INTO @end;
    SET @timeGuard = @start;

#     WHILE (@timeGuard < @end AND @counter <= number) DO
    WHILE (ADDTIME(@timeGuard, '1:00:00') < @end AND @counter <= number) DO

      SELECT id_usługi FROM uslugi WHERE id_wykonawca = @doctor ORDER BY
RAND() LIMIT 1 into @treatment;
      SET @datevalue = CONCAT(@date, ' ', @timeGuard);
      SET @patient = FLOOR(RAND() * (@count_patient - 1)) + 1;
      INSERT INTO wizyty(lekarz, pacjent, gabinet, data_wizyty,
czas_wizyty, id_usługi, priorytet, status_wizyty)
      VALUES (@doctor, @patient, @doctor, @datevalue, '60', @treatment, 1,
'oczekująca');

      SET @timeGuard = ADDTIME(@timeGuard, '1:00:00');
      SET @counter = @counter + 1;
    END WHILE ;

#     Set @date = DATE_ADD(@date, INTERVAL 1 DAY);
    SET @doctor = @doctor + 1;

    IF (@doctor not in (SELECT id_lekarz FROM lekarz)) THEN
      SET @doctor = @doctor + 1;
    END IF ;

    IF (@doctor > @count_doctor) THEN
      SET @doctor = @min_doctor;
    END IF ;

    if @doctor = @min_doctor then
      Set @date = DATE_ADD(@date, INTERVAL 1 DAY);
      WHILE DAYNAME(@date) IN ('Saturday', 'Sunday') DO
        Set @date = DATE_ADD(@date, INTERVAL 1 DAY);
      END WHILE;
    end if;

  END WHILE;
END $$
DELIMITER ;
```