Triggery

PACJENT

```
#trigger przed dodawaniem pracowników sprawdzający mnóstwo rzeczy, np.
- czy data urodzenia zgadza się z peselem
- czy suma po wymnożeniu 10 cyfr peselu przez konkretne wartości i wzięciu
ostatniej cyfry otrzymanego wyniku jest rorwna cyfrze 11.
- czy data urodzenia nie jest wczesiejsza niż 1910 lub późniejsza niż data
obecna
- jeśli data urodzenia jest pozniejsza niż rok 2000 to dodaje się do
miesiąca urodzenia w peselu 20 czyli np. urodzony w 2004-02-02 ma pesel
zaczynający się od 042202..
- czy długość peselu nie jest różna od 11 cyfr
- czy numer telefonu jest liczbą 9-cyfrową
```

```
DELIMITER $$
CREATE TRIGGER dodawacz_pacjentow
  BEFORE INSERT
  ON pacjent
  FOR EACH ROW
  BEGIN
    DECLARE pesel_int BIGINT;
    DECLARE pesel_rok CHAR(2);
    DECLARE pesel_miesiac INT;
    DECLARE pesel_dzien INT;
    DECLARE rok_int INT;
    DECLARE miesiac_int INT;
    DECLARE dzien_int INT;
    DECLARE suma INT;
    DECLARE suma_string VARCHAR(20);
    SET pesel_int = CONVERT(NEW.Pesel, UNSIGNED);
    SET pesel_rok = SUBSTRING(NEW.Pesel, 1, 2);
    SET pesel_miesiac = SUBSTRING(NEW.Pesel, 3, 2);
    SET pesel_dzien = SUBSTRING(NEW.Pesel, 5, 2);
    SET rok_int = CAST(pesel_rok AS UNSIGNED);
    SET miesiac_int = CAST(pesel_miesiac AS UNSIGNED);
    SET dzien_int = CAST(pesel_dzien AS UNSIGNED);
    SET suma = (9 * CAST(SUBSTRING(NEW.Pesel, 1, 1) AS UNSIGNED)
                + 7 * CAST(SUBSTRING(NEW.Pesel, 2, 1) AS UNSIGNED)
                + 3 * CAST(SUBSTRING(NEW.Pesel, 3, 1) AS UNSIGNED)
                + 1 * CAST(SUBSTRING(NEW.Pesel, 4, 1) AS UNSIGNED)
                + 9 * CAST(SUBSTRING(NEW.Pesel, 5, 1) AS UNSIGNED)
                + 7 * CAST(SUBSTRING(NEW.Pesel, 6, 1) AS UNSIGNED)
                + 3 * CAST(SUBSTRING(NEW.Pesel, 7, 1) AS UNSIGNED)
                + 1 * CAST(SUBSTRING(NEW.Pesel, 8, 1) AS UNSIGNED)
                + 9 * CAST(SUBSTRING(NEW.Pesel, 9, 1) AS UNSIGNED)
                + 7 * CAST(SUBSTRING(NEW.Pesel, 10, 1) AS UNSIGNED));

    SET suma_string = CONCAT(suma, 'X');
```

```
    IF YEAR(NEW.Data_urodzenia) < 1910 OR NEW.Data_urodzenia > CURDATE()
    THEN
      SIGNAL SQLSTATE '45000'
      SET MESSAGE_TEXT = 'Niepoprawna data urodzenia';
    END IF;

    IF YEAR(NEW.Data_urodzenia) < 2000
    THEN
      IF (CHAR_LENGTH(NEW.Pesel) <> 11 OR
          CONCAT('19', pesel_rok) <> YEAR(NEW.Data_urodzenia) OR
          miesiac_int <> MONTH(NEW.Data_urodzenia) OR
          dzien_int <> DAY(NEW.Data_urodzenia)
          #KONTROLA czy ostatnia cyfra otrzymanego wyniku 9×a + 7×b + 3×c +
1×d + 9×e + 7×f + 3×g + 1×h + 9×i + 7×j
          #jest równa 11. cyfrze
          OR SUBSTRING(suma_string, CHAR_LENGTH(suma_string)-1, 1) <>
SUBSTRING(NEW.Pesel, 11, 1)
      )
      THEN
        SIGNAL SQLSTATE '45000'
        SET MESSAGE_TEXT = 'Niepoprawny PESEL';
      END IF;
    ELSE
      IF CHAR_LENGTH(NEW.Pesel) <> 11 OR
         CONCAT('20', pesel_rok) <> YEAR(NEW.Data_urodzenia) OR
         (miesiac_int - 20) <> MONTH(NEW.Data_urodzenia) OR
         dzien_int <> DAY(NEW.Data_urodzenia) OR
         SUBSTRING(suma_string, CHAR_LENGTH(suma_string)-1, 1) <>
SUBSTRING(NEW.Pesel, 11, 1)
      THEN
        SIGNAL SQLSTATE '45000'
        SET MESSAGE_TEXT = 'Niepoprawny PESEL';
      END IF;
    END IF;

    IF CHAR_LENGTH(NEW.nr_telefonu) <> 9
    THEN
      SIGNAL SQLSTATE '45000'
      SET MESSAGE_TEXT = 'Niepoprawny numer telefonu! Powinien zawierać 9
cyfr!';
    END IF;

  END $$
DELIMITER ;
```

```
#trigger przed aktualizowaniem pacjentow, który sprawdza wiele rzeczy
wymienionych powyżej w poprzednim triggerze

DELIMITER $$
CREATE TRIGGER zmiana_pacjentow
  BEFORE UPDATE ON pacjent
  FOR EACH ROW
  BEGIN
    DECLARE pesel_int BIGINT;
    DECLARE pesel_rok CHAR(2);
    DECLARE pesel_miesiac INT;
    DECLARE pesel_dzien INT;
```

```
    DECLARE rok_int INT;
    DECLARE miesiac_int INT;
    DECLARE dzien_int INT;
    DECLARE suma INT;
    DECLARE suma_string VARCHAR(20);
    SET pesel_int = CONVERT(NEW.Pesel, UNSIGNED);
    SET pesel_rok = SUBSTRING(NEW.Pesel, 1, 2);
    SET pesel_miesiac = SUBSTRING(NEW.Pesel, 3, 2);
    SET pesel_dzien = SUBSTRING(NEW.Pesel, 5, 2);
    SET rok_int = CAST(pesel_rok AS UNSIGNED);
    SET miesiac_int = CAST(pesel_miesiac AS UNSIGNED);
    SET dzien_int = CAST(pesel_dzien AS UNSIGNED);
    SET suma = (9 * CAST(SUBSTRING(NEW.Pesel, 1, 1) AS UNSIGNED)
                + 7 * CAST(SUBSTRING(NEW.Pesel, 2, 1) AS UNSIGNED)
                + 3 * CAST(SUBSTRING(NEW.Pesel, 3, 1) AS UNSIGNED)
                + 1 * CAST(SUBSTRING(NEW.Pesel, 4, 1) AS UNSIGNED)
                + 9 * CAST(SUBSTRING(NEW.Pesel, 5, 1) AS UNSIGNED)
                + 7 * CAST(SUBSTRING(NEW.Pesel, 6, 1) AS UNSIGNED)
                + 3 * CAST(SUBSTRING(NEW.Pesel, 7, 1) AS UNSIGNED)
                + 1 * CAST(SUBSTRING(NEW.Pesel, 8, 1) AS UNSIGNED)
                + 9 * CAST(SUBSTRING(NEW.Pesel, 9, 1) AS UNSIGNED)
                + 7 * CAST(SUBSTRING(NEW.Pesel, 10, 1) AS UNSIGNED));

    SET suma_string = CONCAT(suma, 'X');

    IF YEAR(NEW.Data_urodzenia) < 1910 OR NEW.Data_urodzenia > CURDATE()
    THEN
      SIGNAL SQLSTATE '45000'
      SET MESSAGE_TEXT = 'Niepoprawna data urodzenia';
    END IF;

    IF YEAR(NEW.Data_urodzenia) < 2000
    THEN
      IF (CHAR_LENGTH(NEW.Pesel) <> 11 OR
          CONCAT('19', pesel_rok) <> YEAR(NEW.Data_urodzenia) OR
          miesiac_int <> MONTH(NEW.Data_urodzenia) OR
          dzien_int <> DAY(NEW.Data_urodzenia)
          #KONTROLA czy ostatnia cyfra otrzymanego wyniku 9×a + 7×b + 3×c +
1×d + 9×e + 7×f + 3×g + 1×h + 9×i + 7×j
          #jest równa 11. cyfrze
          OR SUBSTRING(suma_string, CHAR_LENGTH(suma_string)-1, 1) <>
SUBSTRING(NEW.Pesel, 11, 1)
      )
      THEN
        SIGNAL SQLSTATE '45000'
        SET MESSAGE_TEXT = 'Niepoprawny PESEL';
      END IF;
    ELSE
      IF CHAR_LENGTH(NEW.Pesel) <> 11 OR
         CONCAT('20', pesel_rok) <> YEAR(NEW.Data_urodzenia) OR
         (miesiac_int - 20) <> MONTH(NEW.Data_urodzenia) OR
         dzien_int <> DAY(NEW.Data_urodzenia) OR
         SUBSTRING(suma_string, CHAR_LENGTH(suma_string)-1, 1) <>
SUBSTRING(NEW.Pesel, 11, 1)
      THEN
        SIGNAL SQLSTATE '45000'
        SET MESSAGE_TEXT = 'Niepoprawny PESEL';
      END IF;
    END IF;

    IF CHAR_LENGTH(NEW.nr_telefonu) <> 9
```

```
    THEN
      SIGNAL SQLSTATE '45000'
      SET MESSAGE_TEXT = 'Niepoprawny numer telefonu! Powinien zawierać 9
cyfr!';
    END IF;

  END $$
DELIMITER ;
```

PRACOWNICY

```
#trigger przed dodawaniem pracowników sprawdzający mnóstwo rzeczy, np.
- czy data urodzenia zgadza się z peselem
- czy suma po wymnożeniu 10 cyfr peselu przez konkretne wartości i wzięciu
ostatniej cyfry otrzymanego wyniku jest rorwna cyfrze 11.
- czy data urodzenia nie jest wczesiejsza niż 1910 lub późniejsza niż data
obecna
- jeśli data urodzenia jest pozniejsza niż rok 2000 to dodaje się do
miesiąca urodzenia w peselu 20 czyli np. urodzony w 2004-02-02 ma pesel
zaczynający się od 042202..
- czy długość peselu nie jest różna od 11 cyfr
- czy wartość pensji pracownika nie jest liczbą ujemną
- czy numer telefonu jest liczbą 9-cyfrową

DELIMITER $$
CREATE TRIGGER dodawacz_pracownikow
  BEFORE INSERT
  ON pracownicy
  FOR EACH ROW
  BEGIN
    DECLARE pesel_int BIGINT;
    DECLARE pesel_rok CHAR(2);
    DECLARE pesel_miesiac INT;
    DECLARE pesel_dzien INT;
    DECLARE rok_int INT;
    DECLARE miesiac_int INT;
    DECLARE dzien_int INT;
    DECLARE suma INT;
    DECLARE suma_string VARCHAR(20);
    SET pesel_int = CONVERT(NEW.Pesel, UNSIGNED);
    SET pesel_rok = SUBSTRING(NEW.Pesel, 1, 2);
    SET pesel_miesiac = SUBSTRING(NEW.Pesel, 3, 2);
    SET pesel_dzien = SUBSTRING(NEW.Pesel, 5, 2);
    SET rok_int = CAST(pesel_rok AS UNSIGNED);
    SET miesiac_int = CAST(pesel_miesiac AS UNSIGNED);
    SET dzien_int = CAST(pesel_dzien AS UNSIGNED);
    SET suma = (9 * CAST(SUBSTRING(NEW.Pesel, 1, 1) AS UNSIGNED)
                + 7 * CAST(SUBSTRING(NEW.Pesel, 2, 1) AS UNSIGNED)
                + 3 * CAST(SUBSTRING(NEW.Pesel, 3, 1) AS UNSIGNED)
                + 1 * CAST(SUBSTRING(NEW.Pesel, 4, 1) AS UNSIGNED)
                + 9 * CAST(SUBSTRING(NEW.Pesel, 5, 1) AS UNSIGNED)
                + 7 * CAST(SUBSTRING(NEW.Pesel, 6, 1) AS UNSIGNED)
                + 3 * CAST(SUBSTRING(NEW.Pesel, 7, 1) AS UNSIGNED)
                + 1 * CAST(SUBSTRING(NEW.Pesel, 8, 1) AS UNSIGNED)
                + 9 * CAST(SUBSTRING(NEW.Pesel, 9, 1) AS UNSIGNED)
                + 7 * CAST(SUBSTRING(NEW.Pesel, 10, 1) AS UNSIGNED));
```

```
    SET suma_string = CONCAT(suma, 'X');


    IF YEAR(NEW.Data_urodzenia) < 1910 OR NEW.Data_urodzenia > CURDATE()
    THEN
      SIGNAL SQLSTATE '45000'
      SET MESSAGE_TEXT = 'Niepoprawna data urodzenia';
    END IF;

    IF YEAR(NEW.Data_urodzenia) < 2000
    THEN
      IF (CHAR_LENGTH(NEW.Pesel) <> 11 OR
          CONCAT('19', pesel_rok) <> YEAR(NEW.Data_urodzenia) OR
          miesiac_int <> MONTH(NEW.Data_urodzenia) OR
          dzien_int <> DAY(NEW.Data_urodzenia)
          #KONTROLA czy ostatnia cyfra otrzymanego wyniku 9×a + 7×b + 3×c +
1×d + 9×e + 7×f + 3×g + 1×h + 9×i + 7×j
          #jest równa 11. cyfrze
          OR SUBSTRING(suma_string, CHAR_LENGTH(suma_string)-1, 1) <>
SUBSTRING(NEW.Pesel, 11, 1)
      )
      THEN
        SIGNAL SQLSTATE '45000'
        SET MESSAGE_TEXT = 'Niepoprawny PESEL';
      END IF;
    ELSE
      IF CHAR_LENGTH(NEW.Pesel) <> 11 OR
         CONCAT('20', pesel_rok) <> YEAR(NEW.Data_urodzenia) OR
         (miesiac_int - 20) <> MONTH(NEW.Data_urodzenia) OR
         dzien_int <> DAY(NEW.Data_urodzenia)
      THEN
        SIGNAL SQLSTATE '45000'
        SET MESSAGE_TEXT = 'Niepoprawny PESEL';
      END IF;
    END IF;

    IF NEW.Pensja < 0
    THEN
      SIGNAL SQLSTATE '45000'
      SET MESSAGE_TEXT = 'Niepoprawna wartość pensji';
    END IF;
    IF CHAR_LENGTH(NEW.nr_telefonu) <> 9
    THEN
      SIGNAL SQLSTATE '45000'
      SET MESSAGE_TEXT = 'Niepoprawny numer telefonu! Powinien zawierać 9
cyfr!';
    END IF;

  END $$
DELIMITER ;
```

#trigger przed aktualizowaniem pracowników sprawdzający mnóstwo rzeczy wymienionych powyżej

```
create trigger zmiana_pracownikow
  before UPDATE
  on pracownicy
  for each row
  BEGIN
    DECLARE pesel_int BIGINT;
    DECLARE pesel_rok CHAR(2);
```

```
    DECLARE pesel_miesiac INT;
    DECLARE pesel_dzien INT;
    DECLARE rok_int INT;
    DECLARE miesiac_int INT;
    DECLARE dzien_int INT;
    DECLARE suma INT;
    DECLARE suma_string VARCHAR(20);
    SET pesel_int = CONVERT(NEW.Pesel, UNSIGNED);
    SET pesel_rok = SUBSTRING(NEW.Pesel, 1, 2);
    SET pesel_miesiac = SUBSTRING(NEW.Pesel, 3, 2);
    SET pesel_dzien = SUBSTRING(NEW.Pesel, 5, 2);
    SET rok_int = CAST(pesel_rok AS UNSIGNED);
    SET miesiac_int = CAST(pesel_miesiac AS UNSIGNED);
    SET dzien_int = CAST(pesel_dzien AS UNSIGNED);
    SET suma = (9 * CAST(SUBSTRING(NEW.Pesel, 1, 1) AS UNSIGNED)
                + 7 * CAST(SUBSTRING(NEW.Pesel, 2, 1) AS UNSIGNED)
                + 3 * CAST(SUBSTRING(NEW.Pesel, 3, 1) AS UNSIGNED)
                + 1 * CAST(SUBSTRING(NEW.Pesel, 4, 1) AS UNSIGNED)
                + 9 * CAST(SUBSTRING(NEW.Pesel, 5, 1) AS UNSIGNED)
                + 7 * CAST(SUBSTRING(NEW.Pesel, 6, 1) AS UNSIGNED)
                + 3 * CAST(SUBSTRING(NEW.Pesel, 7, 1) AS UNSIGNED)
                + 1 * CAST(SUBSTRING(NEW.Pesel, 8, 1) AS UNSIGNED)
                + 9 * CAST(SUBSTRING(NEW.Pesel, 9, 1) AS UNSIGNED)
                + 7 * CAST(SUBSTRING(NEW.Pesel, 10, 1) AS UNSIGNED));

    SET suma_string = CONCAT(suma, 'X');


    IF YEAR(NEW.Data_urodzenia) < 1910 OR NEW.Data_urodzenia > CURDATE()
    THEN
      SIGNAL SQLSTATE '45000'
      SET MESSAGE_TEXT = 'Niepoprawna data urodzenia';
    END IF;

    IF YEAR(NEW.Data_urodzenia) < 2000
    THEN
      IF (CHAR_LENGTH(NEW.Pesel) <> 11 OR
          CONCAT('19', pesel_rok) <> YEAR(NEW.Data_urodzenia) OR
          miesiac_int <> MONTH(NEW.Data_urodzenia) OR
          dzien_int <> DAY(NEW.Data_urodzenia)
          OR SUBSTRING(suma_string, CHAR_LENGTH(suma_string)-1, 1) <>
SUBSTRING(NEW.Pesel, 11, 1)
      )
      THEN
        SIGNAL SQLSTATE '45000'
        SET MESSAGE_TEXT = 'Niepoprawny PESEL';
      END IF;
    ELSE
      IF CHAR_LENGTH(NEW.Pesel) <> 11 OR
         CONCAT('20', pesel_rok) <> YEAR(NEW.Data_urodzenia) OR
         (miesiac_int - 20) <> MONTH(NEW.Data_urodzenia) OR
         dzien_int <> DAY(NEW.Data_urodzenia)
      THEN
        SIGNAL SQLSTATE '45000'
        SET MESSAGE_TEXT = 'Niepoprawny PESEL';
      END IF;
    END IF;

    IF NEW.Pensja < 0
    THEN
      SIGNAL SQLSTATE '45000'
```

```
      SET MESSAGE_TEXT = 'Niepoprawna wartość pensji';
    END IF;
    IF CHAR_LENGTH(NEW.nr_telefonu) <> 9
    THEN
      SIGNAL SQLSTATE '45000'
      SET MESSAGE_TEXT = 'Niepoprawny numer telefonu! Powinien zawierać 9
cyfr!';
    END IF;

  END;
```

USŁUGI

```
#Trigger przed UPDATE usługi zeby mozna bylo zapobiec wlasnie utworzeniu
tej samej uslugi dla tego samego lekarza

DELIMITER $$
CREATE TRIGGER aktualizujUsluge BEFORE UPDATE ON uslugi
FOR EACH ROW
BEGIN
  IF(NEW.id_wykonawca IN (SELECT DISTINCT id_wykonawca FROM uslugi)) THEN
    IF(NEW.zabieg_id IN (SELECT zabieg_id FROM uslugi WHERE id_wykonawca =
NEW.id_wykonawca)) THEN
      SIGNAL SQLSTATE '45000'
      SET MESSAGE_TEXT = 'Ten lekarz ma już dodaną tą usługę w określonej
cenie!';
      END IF;
    END IF;
END $$
DELIMITER ;
```

```
#Trigger przed dodawaniem nowych usług, żeby nie
#można bylo dodac kolejnej tej samej usługi dla tego samego lekarza w innej
cenie skoro już istnieje

DELIMITER $$
CREATE TRIGGER dodajUsluge BEFORE INSERT ON uslugi
FOR EACH ROW
BEGIN
  IF(NEW.id_wykonawca IN (SELECT DISTINCT id_wykonawca FROM uslugi)) THEN
    IF(NEW.zabieg_id IN (SELECT zabieg_id FROM uslugi WHERE id_wykonawca =
NEW.id_wykonawca)) THEN
      SIGNAL SQLSTATE '45000'
      SET MESSAGE_TEXT = 'Ten lekarz ma już dodaną tą usługę w określonej
cenie!';
      END IF;
    END IF;
END $$
DELIMITER ;
```

DYŻURY

```
# Trigger przed dodawaniem dyżurów, żeby nie mozna dodać dla lekarza dyzuru
w dany dzień, jak juz ma go ustalonego oraz  sprawdzamy, czy początek
dużury jest wcześniejszy niż koniec dyżuru oraz czy dzień dyżuru jest dniem
tygodnia od poniedziałku do piątku, ponieważ w sobote i w niedzielę jest
nieczynne.

DELIMITER $$
CREATE TRIGGER do_wizyt BEFORE INSERT ON dyzury
FOR EACH ROW
BEGIN
  IF(NEW.dzien IN (SELECT dzien FROM dyzury WHERE id_lekarz =
NEW.id_lekarz)) THEN
      SIGNAL SQLSTATE '45000'
      SET MESSAGE_TEXT = 'Lekarz ma już w ten dzień ustalony swoje godziny
dyżuru!';
      END IF;
  IF(NEW.poczatek > NEW.koniec) THEN
      SIGNAL SQLSTATE '45000'
      SET MESSAGE_TEXT = 'Niepoprawny przedział czasowy dyżuru!';
      END IF;
  IF(NEW.dzien NOT IN ('Monday','Tuesday', 'Wednesday', 'Thursday',
'Friday')) THEN
    SIGNAL SQLSTATE '45000'
      SET MESSAGE_TEXT = 'Niepoprawny dzień dyżuru!';
      END IF;
END $$
DELIMITER ;


# Trigger przed aktualizowaniem dyżurów, żeby nie mozna dodać dla lekarza
dyzuru w dany dzień, jak juz ma go ustalonego oraz  sprawdzamy, czy
początek dużury jest wcześniejszy niż koniec dyżuru oraz czy dzień dyżuru
jest dniem tygodnia od poniedziałku do piątku, ponieważ w sobote i w
niedzielę jest nieczynne.


DELIMITER $$
CREATE TRIGGER aktualizacja_dyzurow BEFORE UPDATE ON dyzury
FOR EACH ROW
BEGIN
  IF(NEW.dzien IN (SELECT dzien FROM dyzury WHERE id_lekarz =
NEW.id_lekarz)) THEN
      SIGNAL SQLSTATE '45000'
      SET MESSAGE_TEXT = 'Lekarz ma już w ten dzień ustalony swoje godziny
dyżuru!';
      END IF;
  IF(NEW.poczatek > NEW.koniec) THEN
      SIGNAL SQLSTATE '45000'
      SET MESSAGE_TEXT = 'Niepoprawny przedział czasowy dyżuru!';
      END IF;
  IF(NEW.dzien NOT IN ('Monday','Tuesday', 'Wednesday', 'Thursday',
'Friday')) THEN
    SIGNAL SQLSTATE '45000'
      SET MESSAGE_TEXT = 'Niepoprawny dzień dyżuru!';
```

```
      END IF;
END $$
DELIMITER ;
```

URLOPY

```
# Trigger do zmieniania statusu wizyt oczekujących na „do przełożenia”, po przyjęciu przez dyrektora urlopu dla danego pracownika

DELIMITER $$
CREATE TRIGGER po_wzieciu_urlopu AFTER UPDATE ON urlopy
FOR EACH ROW
BEGIN
  IF (NEW.status = 'przyjęty') THEN
    UPDATE wizyty SET status_wizyty = 'do przełożenia'
    WHERE status_wizyty = 'oczekująca' AND lekarz = OLD.pracownik
    AND data_wizyty BETWEEN OLD.początek_urlopu AND OLD.koniec_urlopu;
  END IF;
END $$
DELIMITER ;
```

```
# Trigger do sprawdzania poprawności danych urlopu przed aktualizowaniem, czy początek urlopu nie jest wcześniejszy niż obecna data i czy początek urlopu nie jest później niż koniec.

create trigger aktualizujUrlop
  before UPDATE
  on urlopy
  for each row
  BEGIN
  IF(NEW.początek_urlopu < CURDATE() OR
DATEDIFF(NEW.koniec_urlopu,NEW.początek_urlopu)<0) THEN
      SIGNAL SQLSTATE '45000'
      SET MESSAGE_TEXT = 'Niepoprawna data urlopu!';
      END IF;
END;
```

```
# Trigger przed dodawaniem urlopów
DELIMITER $$
CREATE TRIGGER dodajUrlop BEFORE INSERT ON urlopy
FOR EACH ROW
BEGIN
  IF(NEW.początek_urlopu < CURDATE() OR
DATEDIFF(NEW.koniec_urlopu,NEW.początek_urlopu)<0) THEN
      SIGNAL SQLSTATE '45000'
      SET MESSAGE_TEXT = 'Niepoprawna data urlopu!';
      END IF;
END $$
DELIMITER ;
```

WIZYTY

```
#przewidziane są tylko do zapisywania w historii zmiany rekordów w tabeli
wizyty
#gdzie zmieniła się data wizyty lub lekarz. zmiana gabinetu, celu wizyty
itp. nie jest zapisywana w logach.

DELIMITER $$
CREATE TRIGGER historia_zmian_wizyt AFTER UPDATE ON wizyty
FOR EACH ROW
BEGIN
IF OLD.data_wizyty <> NEW.data_wizyty OR OLD.lekarz <> NEW.lekarz
THEN
INSERT INTO logi (wizyta_id, Poprzednia_data_wizyty, Nowa_data_wizyty,
Poprzedni_lekarz, Nowy_lekarz, Data_zmiany)
VALUES(OLD.id_wizyty, OLD.data_wizyty, NEW.data_wizyty, OLD.lekarz,
NEW.lekarz, CURRENT_DATE);
END IF;
IF OLD.pacjent <> NEW.pacjent OR OLD.id_wizyty <> NEW.id_wizyty THEN
SIGNAL SQLSTATE '45000'
SET MESSAGE_TEXT = 'Nie możesz zmienić tych danych wizyty!';
END IF;
END $$
DELIMITER ;
```

```
# Trigger przed dodawaniem wizyt, który sprawdza ich poprawność np. to
- czy data nowej wizyty nie jest wczesniejsza niż obecna data,
- czy dniem wizyty nie jest sobota i niedziela, gdyż w te dni przychodnia
jest nieczynna,
-czy usługa wykonana w trakcie wizyty zawiera się w dostępnych usługach dla
danego lekarza, który ją wykonał,
- czy dany lekarz do którego recepcjonistka zapisuje wizytę na ten dzień
nie ma akurat wziętego urlopu
- czy godzina wizyty jest zgodna z dyżurem lekarza na ten dany dzień, w
który jest ustalana wizyta (np. jeśli ma dyzur od 10:00 do 15:00 to wizyta
nie może być ustalona na 18:00)
- czy już w tym czasie nie jest umówiona jakaś wizyta

DELIMITER $$
CREATE TRIGGER sprawdzanie_wizyt BEFORE INSERT ON wizyty
FOR EACH ROW
BEGIN
  IF(NEW.data_wizyty < CURRENT_TIMESTAMP()) THEN
      SIGNAL SQLSTATE '45000'
      SET MESSAGE_TEXT = 'Niepoprawna data wizyty!';
  END IF;

  IF(DAYNAME(NEW.data_wizyty) IN ('Saturday', 'Sunday')) THEN
    SIGNAL SQLSTATE '45000'
    SET MESSAGE_TEXT = 'Nie można dodać wizyty w sobotę ani niedzielę!';
  END IF;

  IF(NEW.gabinet > 50) THEN
    SIGNAL SQLSTATE '45000'
    SET MESSAGE_TEXT = 'Zły numer gabinetu';
  END IF;
```

```
  IF(NEW.id_uslugi NOT IN (SELECT id_uslugi FROM uslugi WHERE id_wykonawca
= NEW.lekarz)) THEN
    SIGNAL SQLSTATE '45000'
    SET MESSAGE_TEXT = 'Ten lekarz nie wykonuje takiego zabiegu!';
  END IF;

  IF(NEW.lekarz IN (SELECT pracownik FROM urlopy WHERE status = 'przyjęty'
    AND NEW.data_wizyty BETWEEN początek_urlopu AND koniec_urlopu)) THEN
    SIGNAL SQLSTATE '45000'
    SET MESSAGE_TEXT = 'Ten lekarz wziął tego dnia urlop!';
  END IF;

  IF !(TIME(NEW.data_wizyty) BETWEEN
  (SELECT poczatek FROM dyzury WHERE dzien = DAYNAME(NEW.data_wizyty)
  AND dyzury.id_lekarz = NEW.lekarz) AND (SELECT koniec FROM dyzury WHERE
dzien = DAYNAME(NEW.data_wizyty)
  AND dyzury.id_lekarz = NEW.lekarz)) THEN
    SIGNAL SQLSTATE '45000'
    SET MESSAGE_TEXT = 'Ten lekarz nie ma o tej godzinie dyżuru';
  END IF ;

  IF (EXISTS(SELECT id_wizyty
        FROM wizyty WHERE lekarz = NEW.lekarz AND
        NEW.data_wizyty BETWEEN data_wizyty AND DATE_ADD(data_wizyty,
INTERVAL 60 MINUTE) LIMIT 1)) THEN
    SIGNAL SQLSTATE '45000'
    SET MESSAGE_TEXT = 'W tym czasie jest już umówiona wizyta do tego
lekarza.';
  end if;

END $$
DELIMITER ;
```

```
#Trigger przed aktualizowaniem wizyt

CREATE TRIGGER spr_zmianWizyt BEFORE UPDATE ON wizyty
FOR EACH ROW
BEGIN
  IF(NEW.data_wizyty <> OLD.data_wizyty NEW.data_wizyty <
CURRENT_TIMESTAMP()) THEN
      SIGNAL SQLSTATE '45000'
      SET MESSAGE_TEXT = 'Niepoprawna data wizyty!';
  END IF;

  IF(DAYNAME(NEW.data_wizyty) IN ('Saturday', 'Sunday')) THEN
    SIGNAL SQLSTATE '45000'
    SET MESSAGE_TEXT = 'Nie można dodać wizyty w sobotę ani niedzielę!';
  END IF;

  IF(NEW.gabinet > 50) THEN
    SIGNAL SQLSTATE '45000'
    SET MESSAGE_TEXT = 'Zły numer gabinetu';
  END IF;

  IF(NEW.id_uslugi NOT IN (SELECT id_uslugi FROM uslugi WHERE id_wykonawca
= NEW.lekarz)) THEN
    SIGNAL SQLSTATE '45000'
```

```
    SET MESSAGE_TEXT = 'Ten lekarz nie wykonuje takiego zabiegu!';
  END IF;

  IF(NEW.lekarz IN (SELECT pracownik FROM urlopy WHERE status = 'przyjęty'
    AND NEW.data_wizyty BETWEEN początek_urlopu AND koniec_urlopu)) THEN
    SIGNAL SQLSTATE '45000'
    SET MESSAGE_TEXT = 'Ten lekarz wziął tego dnia urlop!';
  END IF;

  IF !(TIME(NEW.data_wizyty) BETWEEN
  (SELECT poczatek FROM dyzury WHERE dzien = DAYNAME(NEW.data_wizyty)
  AND dyzury.id_lekarz = NEW.lekarz) AND (SELECT koniec FROM dyzury WHERE
dzien = DAYNAME(NEW.data_wizyty)
  AND dyzury.id_lekarz = NEW.lekarz)) THEN
    SIGNAL SQLSTATE '45000'
    SET MESSAGE_TEXT = 'Ten lekarz nie ma o tej godzinie dyżuru';
  END IF ;

  IF ( NEW.data_wizyty <> OLD.data_wizyty EXISTS(SELECT id_wizyty
        FROM wizyty WHERE lekarz = NEW.lekarz AND
        NEW.data_wizyty BETWEEN data_wizyty AND DATE_ADD(data_wizyty,
INTERVAL 60 MINUTE) LIMIT 1)) THEN
    SIGNAL SQLSTATE '45000'
    SET MESSAGE_TEXT = 'W tym czasie jest już umówiona wizyta do tego
lekarza.';
  end if;

END $$
DELIMITER ;
```