Lab 2: queries

Task 1:

```
CREATE DATABASE `db-aparaty`;
```

```
USE db-aparaty;
```

Option 1:

```
CREATE USER IF NOT EXISTS '268442'@'localhost' IDENTIFIED BY 'Jakub442';
```

Option 2:

```
CREATE USER IF NOT EXISTS '268442'@'localhost';
SET PASSWORD FOR '268442'@'localhost' = PASSWORD('Jakub442');
```

```
GRANT SELECT, INSERT, UPDATE ON `db-aparaty`.* TO '268442'@'localhost';
```

```
FLUSH PRIVILEGES;
```

Task 2:

```
CREATE TABLE IF NOT EXISTS Producent (
    ID INT AUTO_INCREMENT,
    nazwa VARCHAR(50),
    kraj VARCHAR(20),
    PRIMARY KEY(ID)
);
```

```
CREATE TABLE IF NOT EXISTS Matryca (
    ID INT AUTO_INCREMENT,
    przekatna DECIMAL(4,2) CHECK(przekatna >= 0),
    rozdzielczosc DECIMAL(3,1) CHECK(rozdzielczosc >= 0),
    typ VARCHAR(10),
    PRIMARY KEY(ID)
);

ALTER TABLE Matryca AUTO_INCREMENT = 100;
```

```
CREATE TABLE IF NOT EXISTS Obiektyw (
    ID INT AUTO_INCREMENT,
    model VARCHAR(30),
    minPrzeslona FLOAT CHECK(minPrzeslona >= 0 AND minPrzeslona <
maxPrzeslona),
    maxPrzeslona FLOAT CHECK(maxPrzeslona > 0),
    PRIMARY KEY(ID)
);
```

```
CREATE TABLE IF NOT EXISTS Aparat (
    model VARCHAR(30),
    producent INT CHECK(producent >= 0),
    matryca INT CHECK(matryca >= 100),
    obiektyw INT CHECK(obiektyw >= 0),
    typ enum('kompaktowy', 'lustrzanka', 'profesjonalny', 'inny'),
    PRIMARY KEY(model),
    CONSTRAINT fk_producent
    FOREIGN KEY(producent)
        REFERENCES Producent(ID),
    CONSTRAINT fk_matryca
    FOREIGN KEY(matryca)
        REFERENCES Matryca(ID),
    CONSTRAINT fk_obiektyw
    FOREIGN KEY(obiektyw)
        REFERENCES Obiektyw(ID)
);
```

Task 3:

If already in mariadb:

```
SYSTEM mariadb -u 268442 -p;
```

Else:

```
mariadb -u 268442;
```

Select `db-aparaty` database:

```
USE db-aparaty;
```

To check the current user:

```
SELECT USER();
```

To check current user's privileges:

```
SHOW GRANTS;
```

Adding records to the tables:

```
INSERT INTO
    Producent(nazwa, kraj)
VALUES
    ('producent_1', 'Chiny'),
    ('producent_2', 'Chiny'),
    ('producent_3', 'Chiny'),
    ('producent_4', 'Chiny'),
    ('producent_5', 'Chiny'),
    ('producent_6', 'Japonia'),
    ('producent_7', 'Japonia'),
    ('producent_8', 'Japonia'),
    ('producent_9', 'Japonia'),
    ('producent_10', 'Japonia'),
    ('producent_11', 'USA'),
    ('producent_12', 'USA'),
    ('producent_13', 'USA'),
    ('producent_14', 'USA'),
    ('producent_15', 'USA');
```

```
INSERT INTO Producent(nazwa, kraj) VALUES
('01234567890123456789012345678901234567890123456789012345678901234567890123', 'kraj_test');

INSERT INTO Producent(nazwa, kraj) VALUES ('producent_test',
'012345678901234567890123');
```

```
INSERT INTO
    Matryca(przekatna, rozdzielczosc, typ)
VALUES
    (1, 1, 'typ_a'),
    (2, 2, 'typ_b'),
    (3, 3, 'typ_c'),
    (1, 4, 'typ_a'),
    (2, 5, 'typ_b'),
    (3, 1, 'typ_c'),
    (1, 2, 'typ_a'),
    (2, 3, 'typ_b'),
    (3, 4, 'typ_c'),
    (1, 5, 'typ_a'),
    (2, 1, 'typ_b'),
    (3, 2, 'typ_c'),
    (1, 3, 'typ_a'),
    (2, 4, 'typ_b'),
    (3, 5, 'typ_c');
```

```
INSERT INTO Matryca(przekatna, rozdzielczosc, typ) VALUES (-1, 1, 'typ_d');

INSERT INTO Matryca(przekatna, rozdzielczosc, typ) VALUES (1, -1, 'typ_d');

INSERT INTO Matryca(przekatna, rozdzielczosc, typ) VALUES (123456.123456,
1, 'typ_d');

INSERT INTO Matryca(przekatna, rozdzielczosc, typ) VALUES (1,
123456.123456, 'typ_d');
```

```
INSERT INTO
    Obiektyw(model, minPrzeslona, maxPrzeslona)
VALUES
    ('model_a', 1.1, 1.2),
    ('model_b1', 1.1, 1.3),
    ('model_c1', 1.1, 1.4),
```

```
    ('model_d1', 1.1, 1.5),
    ('model_e1', 1.1, 1.6),
    ('model_b2', 1.2, 1.3),
    ('model_c2', 1.2, 1.4),
    ('model_d2', 1.2, 1.5),
    ('model_e2', 1.2, 1.6),
    ('model_c3', 1.3, 1.4),
    ('model_d3', 1.3, 1.5),
    ('model_e3', 1.3, 1.6),
    ('model_d4', 1.4, 1.5),
    ('model_e4', 1.4, 1.6),
    ('model_e5', 1.5, 1.6);
```

```
INSERT INTO Obiektyw(model, minPrzeslona, maxPrzeslona) VALUES
('model_test', -1, 1);

INSERT INTO Obiektyw(model, minPrzeslona, maxPrzeslona) VALUES
('model_test', 1, -1);

INSERT INTO Obiektyw(model, minPrzeslona, maxPrzeslona) VALUES
('model_test', 2, 1);
```

```
INSERT INTO
    Aparat(model, producent, matryca, obiektyw, typ)
VALUES
    ('model_0001', 1, 100, 1, 'kompaktowy'),
    ('model_0002', 2, 101, 2, 'kompaktowy'),
    ('model_0003', 3, 102, 3, 'kompaktowy'),
    ('model_0004', 4, 103, 4, 'kompaktowy'),
    ('model_0005', 5, 104, 5, 'lustrzanka'),
    ('model_0006', 6, 105, 6, 'lustrzanka'),
    ('model_0007', 7, 106, 7, 'lustrzanka'),
    ('model_0008', 8, 107, 8, 'lustrzanka'),
    ('model_0009', 9, 108, 9, 'profesjonalny'),
    ('model_0010', 10, 109, 10, 'profesjonalny'),
    ('model_0011', 11, 110, 11, 'profesjonalny'),
    ('model_0012', 12, 111, 12, 'profesjonalny'),
    ('model_0013', 13, 112, 13, 'inny'),
    ('model_0014', 14, 113, 14, 'inny'),
    ('model_0015', 15, 114, 15, 'inny');
```

```
INSERT INTO Aparat(model, producent, matryca, obiektyw, typ) VALUES
('model_test', -1, 100, 1, 'inny');

INSERT INTO Aparat(model, producent, matryca, obiektyw, typ) VALUES
```

```
('model_test', 1, 1, 1, 'inny');

INSERT INTO Aparat(model, producent, matryca, obiektyw, typ) VALUES
('model_test', 1, -100, 1, 'inny');

INSERT INTO Aparat(model, producent, matryca, obiektyw, typ) VALUES
('model_test', 1, 100, -1, 'inny');

INSERT INTO Aparat(model, producent, matryca, obiektyw, typ) VALUES
('model_test', 1, 100, 1, 'abrakadabra');
```

Task 4:

Użytkownik utworzony w `Task 1` nie będzie mógł wywołać procedury generateCameras() ani nie mógłby takiej procedury stworzyć.

```
DELIMITER $$
CREATE PROCEDURE generateCameras()
BEGIN
    DECLARE i INT DEFAULT 0;

    DECLARE curr_model INT DEFAULT 0;
    DECLARE model_name VARCHAR(30) DEFAULT 'model_0000';

    DECLARE rand_producent INT DEFAULT 1;
    DECLARE rand_matryca INT DEFAULT 100;
    DECLARE rand_obiektyw INT DEFAULT 1;

    DECLARE num INT DEFAULT 4;
    DECLARE rand_typ VARCHAR(20) DEFAULT 'inny';

    SET curr_model = (SELECT (RIGHT(MAX(model), 4) + 1) FROM Aparat WHERE
model LIKE 'model_%');

    WHILE i < 100 DO
        SET model_name = (SELECT CONCAT('model_', RIGHT(CONCAT('0000',
CAST(CAST(curr_model AS INT) AS VARCHAR(4))), 4)));
        SET curr_model = curr_model + 1;

        SET rand_producent = (SELECT ID from Producent ORDER BY RAND()
LIMIT 1);
        SET rand_matryca = (SELECT ID from Matryca ORDER BY RAND() LIMIT
1);
        SET rand_obiektyw = (SELECT ID from Obiektyw ORDER BY RAND() LIMIT
1);

        SET num = CEILING(RAND()*4);
```

```
        SET rand_typ = (SELECT ELT(num, 'kompaktowy', 'lustrzanka',
'profesjonalny', 'inny'));

        INSERT INTO Aparat(model, producent, matryca, obiektyw, typ) VALUES
(model_name, rand_producent, rand_matryca, rand_obiektyw, rand_typ);

        SET i = i + 1;
    END WHILE;
END$$

DELIMITER ;
```

```
CALL generateCameras;
```

Task 5:

```
DELIMITER $$
CREATE PROCEDURE maxPrzekatna(id_producenta INT)
BEGIN
    IF EXISTS (SELECT ID FROM Producent WHERE ID = id_producenta) THEN
        SELECT a.model
            FROM Aparat AS a JOIN Matryca AS m ON a.matryca = m.ID
            WHERE a.producent = id_producenta
            ORDER BY m.przekatna DESC
            LIMIT 1;
    ELSE
        SELECT 'Variable id_producenta out of range!' as 'Error!';
    END IF;
END$$

DELIMITER ;
```

```
CALL maxPrzekatna(id);
```

Task 6:

```
DELIMITER $$
CREATE TRIGGER cameraNewProducer
    BEFORE INSERT
    ON Aparat
    FOR EACH ROW

    BEGIN
        IF NOT EXISTS (SELECT ID FROM Producent WHERE ID = NEW.producent)
THEN
            INSERT INTO Producent(ID, nazwa, kraj) VALUES (NEW.producent,
NULL, NULL);
        END IF;
    END$$

DELIMITER ;
```

Task 7:

```
CREATE FUNCTION nCameras (id_matrycy INT)
    RETURNS INT
    RETURN
        (SELECT COUNT(model) FROM Aparat WHERE matryca = id_matrycy);
```

```
SELECT nCameras(id_matrycy);
```

Task 8:

```
DELIMITER $$
CREATE TRIGGER deleteMatrix
    AFTER DELETE
    ON Aparat
    FOR EACH ROW

    BEGIN
        IF NOT EXISTS (SELECT model FROM Aparat WHERE matryca =
OLD.matryca) THEN
            DELETE FROM Matryca WHERE ID = OLD.Matryca;
```

```
        END IF;
    END$$

DELIMITER ;
```

Task 9:

Użytkownik utworzony w `Task 1` nie może stworzyć widoku `nonChineseCamerasData` (ani żadnego innego widoku).

```
CREATE VIEW nonChineseCamerasData AS
    (SELECT a.model, p.nazwa, m.przekatna, m.rozdzielczosc, o.minPrzeslona,
o.maxPrzeslona
        FROM Aparat AS a
        JOIN Producent AS p ON a.producent = p.ID
        JOIN Matryca AS m ON a.matryca = m.ID
        JOIN Obiektyw AS o ON a.obiektyw = o.ID
        WHERE a.typ = 'lustrzanka' AND p.kraj <> 'Chiny');
```

Task 10:

Po usunięciu aparatów od producentów z siedzibą w Chinach z tabeli `Aparat` z widoku `cameraProducer` zostały usunięte wszystkie rekordy, dla których kraj producenta to Chiny.

```
CREATE VIEW cameraProducer AS
    (SELECT a.model, p.nazwa, p.kraj
        FROM Aparat AS a
        JOIN Producent AS p ON a.producent = p.ID);
```

```
DELETE FROM Aparat WHERE producent IN
    (SELECT ID FROM Producent WHERE kraj = 'Chiny');
```

Task 11:

Użytkownik stworzony w `Task 1` nie może utworzyć trigerrów: `producerIncrementCount`, `producerDecrementCount`, `producerUpdateCount` (ani żadnych innych trigerrów), jednak jeżeli te trigerry zostały stworzone przez roota, to będą one działały dla tego użytkownika.

```
ALTER TABLE Producent ADD liczbaModeli INT NOT NULL DEFAULT 0
CHECK(liczbaModeli >= 0);

UPDATE Producent AS p SET p.liczbaModeli = (SELECT COUNT(a.model) FROM
Aparat AS a WHERE a.producent = p.ID);
```

```
DELIMITER $$
CREATE TRIGGER producerIncrementCount
    AFTER INSERT
    ON Aparat
    FOR EACH ROW

    BEGIN
        UPDATE Producent SET liczbaModeli = liczbaModeli + 1 WHERE ID =
NEW.producent;
    END$$


CREATE TRIGGER producerDecrementCount
    AFTER DELETE
    ON Aparat
    FOR EACH ROW

    BEGIN
        UPDATE Producent SET liczbaModeli = liczbaModeli - 1 WHERE ID =
OLD.producent;
    END$$


CREATE TRIGGER producerUpdateCount
    AFTER UPDATE
    ON Aparat
    FOR EACH ROW

    BEGIN
        UPDATE Producent SET liczbaModeli = liczbaModeli + 1 WHERE ID =
NEW.producent;
        UPDATE Producent SET liczbaModeli = liczbaModeli - 1 WHERE ID =
OLD.producent;
    END$$

DELIMITER ;
```