Lab 2

Rafał Włodarczyk

2024-11-16

MariaDB Setup:

```
services:
  mariadb:
    image: mariadb:latest
    container_name: mariadb-container
    environment:
      MYSQL_ROOT_PASSWORD: rootpassword
      MYSQL_DATABASE: db2024
      MYSQL_USER: rafisto
      MYSQL_PASSWORD: rafistopassword
    volumes:
      - db_data:/var/lib/mysql

    ports:
      - "3306:3306"
    networks:
      - mariadb-network

volumes:
  db_data:

networks:
  mariadb-network:
    driver: bridge
```

```
# Maintenance

## Up DB
docker compose up -d
## Down DB
docker compose down && docker volume rm lista2_db_data

# Execution

## Root user
docker exec -it mariadb-container mariadb -D aparaty -u root -prootpassword
## Normal user
docker exec -it mariadb-container mariadb -D aparaty -u 123456 -pkowalski56
```

Solves

Exercise 1

Create db named `aparaty`;

```
CREATE DATABASE aparaty;
```

Create user named `123456` with password `kowalski56`;

```
CREATE USER '123456'@'%' IDENTIFIED BY 'kowalski56';
```

Grant select, insert and update privileges on `aparaty` to `123456`;

```
GRANT SELECT, INSERT, UPDATE ON aparaty.* TO '123456'@'%';
FLUSH PRIVILEGES;
```

Exercise 2

Create tables:

```
USE aparaty;

CREATE TABLE Matryca (
    ID INT PRIMARY KEY AUTO_INCREMENT,
    przekatna DECIMAL(4,2) CHECK (przekatna >= 0), -- 4 cyfry, 2 po
przecinku
    rozdzielczosc DECIMAL(3,1) CHECK (rozdzielczosc >= 0), -- 3 cyfry, 1 po
przecinku
```

```
    typ VARCHAR(10)
) AUTO_INCREMENT = 100;

CREATE TABLE Obiektyw (
    ID INT PRIMARY KEY AUTO_INCREMENT,
    model VARCHAR(30),
    minPrzeslona FLOAT CHECK (minPrzeslona >= 0),
    maxPrzeslona FLOAT CHECK (maxPrzeslona > minPrzeslona),
    CONSTRAINT check_min_max CHECK (minPrzeslona < maxPrzeslona)
) AUTO_INCREMENT = 1;

CREATE TABLE Producent (
    ID INT PRIMARY KEY AUTO_INCREMENT,
    nazwa VARCHAR(50) NOT NULL,
    kraj VARCHAR(20) DEFAULT 'nieznany',
    adresKorespondencyjny VARCHAR(100) DEFAULT 'nieznany'
) AUTO_INCREMENT = 1;

CREATE TABLE Aparat (
    model VARCHAR(30) PRIMARY KEY,
    producent INT,
    matryca INT,
    obiektyw INT,
    waga FLOAT,
    typ ENUM('kompaktowy', 'lustrzanka', 'profesjonalny', 'inny')
    FOREIGN KEY (producent) REFERENCES Producent(ID),
    FOREIGN KEY (matryca) REFERENCES Matryca(ID),
    FOREIGN KEY (obiektyw) REFERENCES Obiektyw(ID)
);
```

Exercise 3

Connect to the db with the created user and add data to the tables;

```
docker exec -it mariadb-container mariadb -D aparaty -u 123456 -pkowalski56
```

```
INSERT INTO Producent(nazwa,kraj) VALUES ('Canon', 'Japonia');
INSERT INTO Producent(nazwa,kraj) VALUES ('Nikon', 'Japonia');
INSERT INTO Producent(nazwa,kraj) VALUES ('Sony', 'Japonia');
INSERT INTO Producent(nazwa,kraj) VALUES ('Panasonic', 'Japonia');
INSERT INTO Producent(nazwa,kraj) VALUES ('Olympus', 'Japonia');
INSERT INTO Producent(nazwa,kraj) VALUES ('Fujifilm', 'Japonia');
INSERT INTO Producent(nazwa,kraj) VALUES ('Huawei', 'Chiny');
INSERT INTO Producent(nazwa,kraj) VALUES ('Xiaomi', 'Chiny');
INSERT INTO Producent(nazwa,kraj) VALUES ('DJI', 'Chiny');
INSERT INTO Producent(nazwa,kraj) VALUES ('YI Technology', 'Chiny');
INSERT INTO Producent(nazwa,kraj) VALUES ('Zhiyun Technology', 'Chiny');
INSERT INTO Producent(nazwa,kraj) VALUES ('GoPro', 'USA');
INSERT INTO Producent(nazwa,kraj) VALUES ('Kodak', 'USA');
```

```
INSERT INTO Producent(nazwa,kraj) VALUES ('Leica', 'Niemcy');
INSERT INTO Producent(nazwa,kraj) VALUES ('Hasselblad', 'Szwecja');

INSERT INTO Matryca(przekatna, rozdzielczosc, typ) VALUES (1.1, 20.1,
'CMOS');
INSERT INTO Matryca(przekatna, rozdzielczosc, typ) VALUES (1.2, 18.3,
'CCD');
INSERT INTO Matryca(przekatna, rozdzielczosc, typ) VALUES (1.5, 22.0,
'CMOS');
INSERT INTO Matryca(przekatna, rozdzielczosc, typ) VALUES (1.3, 19.7,
'CCD');
INSERT INTO Matryca(przekatna, rozdzielczosc, typ) VALUES (1.4, 21.5,
'CMOS');
INSERT INTO Matryca(przekatna, rozdzielczosc, typ) VALUES (1.6, 23.2,
'CCD');
INSERT INTO Matryca(przekatna, rozdzielczosc, typ) VALUES (1.7, 24.0,
'CMOS');
INSERT INTO Matryca(przekatna, rozdzielczosc, typ) VALUES (1.8, 25.5,
'CCD');
INSERT INTO Matryca(przekatna, rozdzielczosc, typ) VALUES (2.0, 26.8,
'CMOS');
INSERT INTO Matryca(przekatna, rozdzielczosc, typ) VALUES (1.9, 28.3,
'CCD');
INSERT INTO Matryca(przekatna, rozdzielczosc, typ) VALUES (2.1, 29.0,
'CMOS');
INSERT INTO Matryca(przekatna, rozdzielczosc, typ) VALUES (2.2, 30.5,
'LIDAR');
INSERT INTO Matryca(przekatna, rozdzielczosc, typ) VALUES (2.3, 32.1,
'CMOS');
INSERT INTO Matryca(przekatna, rozdzielczosc, typ) VALUES (2.4, 34.0,
'CCD');
INSERT INTO Matryca(przekatna, rozdzielczosc, typ) VALUES (2.5, 36.0,
'CMOS');

INSERT INTO Obiektyw(model, minPrzeslona, maxPrzeslona) VALUES ('Canon EF
50mm f/1.8 STM', 1.8, 22);
INSERT INTO Obiektyw(model, minPrzeslona, maxPrzeslona) VALUES ('Nikon AF-S
50mm f/1.8G', 1.8, 16);
INSERT INTO Obiektyw(model, minPrzeslona, maxPrzeslona) VALUES ('Sony FE
24-70mm f/2.8 GM', 2.8, 22);
INSERT INTO Obiektyw(model, minPrzeslona, maxPrzeslona) VALUES ('Tamron 70-
200mm f/2.8 G2', 2.8, 22);
INSERT INTO Obiektyw(model, minPrzeslona, maxPrzeslona) VALUES ('Sigma 35mm
f/1.4 DG HSM', 1.4, 16);
INSERT INTO Obiektyw(model, minPrzeslona, maxPrzeslona) VALUES ('Canon RF
85mm f/1.2L USM', 1.2, 16);
INSERT INTO Obiektyw(model, minPrzeslona, maxPrzeslona) VALUES ('Nikon Z
24-70mm f/2.8 S', 2.8, 22);
INSERT INTO Obiektyw(model, minPrzeslona, maxPrzeslona) VALUES ('Fujifilm
XF 23mm f/1.4 R', 1.4, 16);
INSERT INTO Obiektyw(model, minPrzeslona, maxPrzeslona) VALUES ('Olympus
M.Zuiko 8 PRO', 2.8, 22);
INSERT INTO Obiektyw(model, minPrzeslona, maxPrzeslona) VALUES ('Leica
Summilux-M 50mmH', 1.4, 16);
```

```
INSERT INTO Obiektyw(model, minPrzeslona, maxPrzeslona) VALUES ('Panasonic
Lumix G 25mm', 1.7, 22);
INSERT INTO Obiektyw(model, minPrzeslona, maxPrzeslona) VALUES ('Tokina AT-
X 11-16mm', 2.8, 22);
INSERT INTO Obiektyw(model, minPrzeslona, maxPrzeslona) VALUES ('Samyang
14mm f/2.8', 2.8, 22);
INSERT INTO Obiektyw(model, minPrzeslona, maxPrzeslona) VALUES ('Zeiss
Loxia 35mm f/2', 2, 22);
INSERT INTO Obiektyw(model, minPrzeslona, maxPrzeslona) VALUES
('Voigtlander 35mm', 1.4, 16);

INSERT INTO Aparat(model, producent, matryca, obiektyw, waga, typ) VALUES
('Canon EOS 5D Mark II', 1, 100, 1, 800, 'lustrzanka');
INSERT INTO Aparat(model, producent, matryca, obiektyw, waga, typ) VALUES
('Canon EOS 5D Mark III', 1, 101, 1, 800, 'lustrzanka');
INSERT INTO Aparat(model, producent, matryca, obiektyw, waga, typ) VALUES
('Canon EOS 5D Mark IV', 1, 102, 1, 800, 'lustrzanka');
INSERT INTO Aparat(model, producent, matryca, obiektyw, waga, typ) VALUES
('Canon EOS 5D Mark V', 1, 103, 1, 800, 'lustrzanka');
INSERT INTO Aparat(model, producent, matryca, obiektyw, waga, typ) VALUES
('Nikon D850', 2, 101, 2, 915, 'lustrzanka');
INSERT INTO Aparat(model, producent, matryca, obiektyw, waga, typ) VALUES
('Sony Alpha 7R IV', 3, 102, 3, 665, 'profesjonalny');
INSERT INTO Aparat(model, producent, matryca, obiektyw, waga, typ) VALUES
('Panasonic Lumix GH5', 4, 103, 4, 725, 'profesjonalny');
INSERT INTO Aparat(model, producent, matryca, obiektyw, waga, typ) VALUES
('Olympus OM-D E-M1 Mark III', 5, 104, 5, 500, 'profesjonalny');
INSERT INTO Aparat(model, producent, matryca, obiektyw, waga, typ) VALUES
('Fujifilm X-T4', 6, 105, 6, 600, 'profesjonalny');
INSERT INTO Aparat(model, producent, matryca, obiektyw, waga, typ) VALUES
('Huawei P40 Pro', 7, 106, 7, 209, 'kompaktowy');
INSERT INTO Aparat(model, producent, matryca, obiektyw, waga, typ) VALUES
('Xiaomi Mi 10 Ultra', 8, 107, 8, 220, 'kompaktowy');
INSERT INTO Aparat(model, producent, matryca, obiektyw, waga, typ) VALUES
('DJI Osmo Pocket', 9, 108, 9, 116, 'kompaktowy');
INSERT INTO Aparat(model, producent, matryca, obiektyw, waga, typ) VALUES
('YI 4K Action Camera', 10, 109, 10, 94, 'kompaktowy');
INSERT INTO Aparat(model, producent, matryca, obiektyw, waga, typ) VALUES
('Zhiyun Crane M2', 11, 110, 11, 500, 'inny');
INSERT INTO Aparat(model, producent, matryca, obiektyw, waga, typ) VALUES
('GoPro HERO10 Black', 12, 111, 12, 153, 'kompaktowy');
INSERT INTO Aparat(model, producent, matryca, obiektyw, waga, typ) VALUES
('Kodak PIXPRO SP360 4K', 13, 112, 13, 196, 'kompaktowy');
INSERT INTO Aparat(model, producent, matryca, obiektyw, waga, typ) VALUES
('Leica M10-R', 14, 113, 14, 660, 'lustrzanka');
INSERT INTO Aparat(model, producent, matryca, obiektyw, waga, typ) VALUES
('Hasselblad X1D II 50C', 15, 114, 15, 725, 'profesjonalny');
```

Incorrect data

```
INSERT INTO Producent(nazwa,kraj) VALUES (NULL, 'Chiny');
INSERT INTO Matryca(przekatna, rozdzielczosc, typ) VALUES (-1.1, 3.6,
```

```
'CMOS');
INSERT INTO Obiektyw(model, minPrzeslona, maxPrzeslona) VALUES
('Voigtlander 35mm', 10, 9);
INSERT INTO Aparat(model, producent, matryca, obiektyw, waga, typ) VALUES
('Hasselblad', 15, 1000, 13, 700, 'profesjonalny');
```

Exercise 4

Back to the root user

```
docker exec -it mariadb-container mariadb -D aparaty -u root -prootpassword
```

Then create a procedure creating random 100 cameras provided the initial inserts.

```
DELIMITER $$

CREATE PROCEDURE GenerateRandomAparatys()
BEGIN
    DECLARE i INT DEFAULT 0;
    DECLARE model_name VARCHAR(255);
    DECLARE random_producer INT;
    DECLARE random_sensor INT;
    DECLARE random_lens INT;
    DECLARE random_weight INT;
    DECLARE random_type ENUM('kompaktowy', 'lustrzanka', 'profesjonalny',
'inny');

    WHILE i < 100 DO
        SET model_name = CONCAT('Model ', i + 1);

        SET random_producer = FLOOR(1 + (RAND() * 15));
        SET random_sensor = FLOOR(100 + (RAND() * 15));
        SET random_lens = FLOOR(1 + (RAND() * 15));

        SET random_weight = FLOOR(300 + (RAND() * 1200));
        SET random_type = ELT(FLOOR(1 + (RAND() * 4)), 'kompaktowy',
'lustrzanka', 'profesjonalny', 'inny');

        IF NOT EXISTS (SELECT 1 FROM Aparat WHERE model = model_name) THEN
            INSERT INTO Aparat (model, producent, matryca, obiektyw, waga,
typ)
            VALUES (model_name, random_producer, random_sensor,
random_lens, random_weight, random_type);
        END IF;

        SET i = i + 1;
    END WHILE;

END$$
```

```
DELIMITER ;
```

We can now call the procedure:

```
CALL GenerateRandomAparatys();
```

The user will not be able to perform the procedure, because it lacks the necessary `EXECUTE` privilege.

The user will see

```
ERROR 1370 (42000): execute command denied to user '123456'@'%' for routine
'aparaty.GenerateRandomAparatys'
```

Exercise 5

Create a function or procedure, which for a provided Producer ID will return the model with the smallest matrix diagonal.

```
DELIMITER $$

CREATE OR REPLACE FUNCTION GetSmallestDiagonalModel(producer_id INT)
RETURNS VARCHAR(255)
BEGIN
    DECLARE model_name VARCHAR(255);

    SELECT Aparat.model
    INTO model_name
    FROM Aparat
    JOIN Matryca ON Aparat.matryca = Matryca.ID
    WHERE Aparat.producent = producer_id
    ORDER BY Matryca.przekatna ASC
    LIMIT 1;

    RETURN model_name;
END$$

DELIMITER ;
```

Use with:

```
SELECT GetSmallestDiagonalModel(1);
```

Exercise 6

```
DELIMITER $$

CREATE TRIGGER InsertProducentBeforeAparat
BEFORE INSERT ON Aparat
FOR EACH ROW
BEGIN
   IF NOT EXISTS (SELECT 1 FROM Producent WHERE ID = NEW.producent) THEN
      INSERT INTO Producent (ID, nazwa) VALUES (NEW.producent, 'Nowy
Producent');
   END IF;
END$$

DELIMITER ;
```

Exercise 7

```
DELIMITER $$

CREATE FUNCTION LiczbaModeli(matrycaID INT) RETURNS INT
BEGIN
   RETURN (SELECT COUNT(*) FROM Aparat WHERE matryca = matrycaID);
END$$

DELIMITER ;
```

Use by querying:

```
SELECT LiczbaModeli(100);
```

Exercise 8

```
DELIMITER $$

CREATE TRIGGER CascadeDeleteMatryca
AFTER DELETE ON Aparat
FOR EACH ROW
BEGIN
   IF NOT EXISTS (SELECT 1 FROM Aparat WHERE matryca = OLD.matryca) THEN
      DELETE FROM Matryca WHERE ID = OLD.matryca;
   END IF;
END$$

DELIMITER ;
```

Exercise 9

The user will not be able to perform the action, because it lacks the `CREATE VIEW` privilege.

```
CREATE VIEW Lustrzankas AS
SELECT Aparat.model, Aparat.waga, Producent.nazwa, Matryca.przekatna,
Matryca.rozdzielczosc, Obiektyw.minPrzeslona, Obiektyw.maxPrzeslona
FROM Aparat
JOIN Producent ON Aparat.producent = Producent.ID
JOIN Matryca ON Aparat.matryca = Matryca.ID
JOIN Obiektyw ON Aparat.obiektyw = Obiektyw.ID
WHERE Aparat.typ = 'lustrzanka' AND Producent.kraj != 'Chiny';
```

Exercise 10

The view will change according to the changes in the tables, as it is a dynamic view.

```
CREATE VIEW CameraProducer AS
SELECT Producent.nazwa, Producent.kraj, Aparat.model
FROM Aparat
JOIN Producent ON Aparat.producent = Producent.ID;

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

Exercise 11

The user will not be able to perform the action, because it lacks the `CREATE TRIGGER` privilege. Trigger works no matter which user has performed the action.

```
ALTER TABLE Producent ADD COLUMN liczbaModeli INT NOT NULL DEFAULT 0;
UPDATE Producent SET liczbaModeli = (SELECT COUNT(*) FROM Aparat WHERE
Aparat.producent = Producent.ID);

DELIMITER $$

CREATE TRIGGER InsertAparat
AFTER INSERT ON Aparat
FOR EACH ROW
BEGIN
    if NEW.producent IS NOT NULL THEN
        UPDATE Producent SET liczbaModeli = liczbaModeli + 1 WHERE ID =
NEW.producent;
    END IF;
END$$

CREATE TRIGGER UpdateAparat
AFTER UPDATE ON Aparat
```

```
FOR EACH ROW
BEGIN
   IF OLD.producent IS NOT NULL THEN
      UPDATE Producent SET liczbaModeli = liczbaModeli - 1 WHERE ID =
OLD.producent;
   END IF;
   IF NEW.producent IS NOT NULL THEN
      UPDATE Producent SET liczbaModeli = liczbaModeli + 1 WHERE ID =
NEW.producent;
   END IF;
END$$

CREATE TRIGGER DeleteAparat
AFTER DELETE ON Aparat
FOR EACH ROW
BEGIN
   IF OLD.producent IS NOT NULL THEN
      UPDATE Producent SET liczbaModeli = liczbaModeli - 1 WHERE ID =
OLD.producent;
   END IF;
END$$

DELIMITER ;
```