1. CREATE DATABASE ludziedb;

```
CREATE TABLE Ludzie (
 lp INT UNSIGNED NOT NULL AUTO_INCREMENT PRIMARY KEY,
 PESEL CHAR(11) CHECK(CHAR_LENGTH(PESEL) = 11),
 imie VARCHAR(30),
 nazwisko VARCHAR(30),
 data_urodzenia DATE,
 plec ENUM('K', 'M')
);

CREATE TABLE Zawody (
 zawod_id INT UNSIGNED NOT NULL AUTO_INCREMENT PRIMARY KEY,
 nazwa VARCHAR(50),
 pensja_min FLOAT CHECK(pensja_min >= 0),
 pensja_max FLOAT CHECK(pensja_max >= 0),
 CHECK(pensja_min < pensja_max)
);

CREATE TABLE Pracownicy (
 lp INT UNSIGNED NOT NULL,
 zawod_id INT UNSIGNED NOT NULL,
 pensja FLOAT CHECK(pensja >= 0),
 FOREIGN KEY (lp) REFERENCES Ludzie(lp),
 FOREIGN KEY (zawod_id) REFERENCES Zawody(zawod_id)
);

DELIMITER $$
CREATE FUNCTION GetPeselMonth(data_ur DATE)
RETURNS VARCHAR(2) DETERMINISTIC
BEGIN
 IF SUBSTR(YEAR(data_ur), 1, 2) = 18 THEN RETURN (80 + MONTH(data_ur));
 ELSEIF SUBSTR(YEAR(data_ur), 1, 2) = 19 THEN RETURN (DATE_FORMAT(data_ur,
'%m'));
 ELSEIF SUBSTR(YEAR(data_ur), 1, 2) = 20 THEN RETURN (20 + MONTH(data_ur));
 ELSEIF SUBSTR(YEAR(data_ur), 1, 2) = 21 THEN RETURN (40 + MONTH(data_ur));
 ELSEIF SUBSTR(YEAR(data_ur), 1, 2) = 22 THEN RETURN (60 + MONTH(data_ur));
 END IF;
END$$
DELIMITER ;

DELIMITER $$
CREATE FUNCTION PeselDataCheck(pesel CHAR(11), data_ur DATE)
RETURNS BOOL DETERMINISTIC
BEGIN
 IF SUBSTR(pesel, 1, 2) <> SUBSTR(YEAR(data_ur), 3, 2) THEN RETURN FALSE;
 ELSEIF SUBSTR(pesel, 3, 2) <> GetPeselMonth(data_ur) THEN RETURN FALSE;
 ELSEIF SUBSTR(pesel, 5, 2) <> DAY(data_ur) THEN RETURN FALSE;
 ELSE RETURN TRUE;
```

```
  END IF;
END$$
DELIMITER ;

DELIMITER $$
CREATE FUNCTION PeselPlecCheck(pesel CHAR(11), plec ENUM('K', 'M'))
RETURNS BOOL DETERMINISTIC
BEGIN
 IF plec = 'K' THEN RETURN SUBSTR(pesel, 10, 1) % 2 = 0;
 ELSE RETURN SUBSTR(pesel, 10, 1) % 2 = 1;
 END IF;
END$$
DELIMITER ;

DELIMITER $$
CREATE FUNCTION PeselControlCheck(pesel CHAR(11))
RETURNS BOOL DETERMINISTIC
BEGIN
 RETURN (10 - (SUBSTR(pesel, 1, 1) * 1 + SUBSTR(pesel, 2, 1) * 3 + SUBSTR(pesel, 3, 1) *
7 + SUBSTR(pesel, 4, 1) * 9 + SUBSTR(pesel, 5, 1) * 1 + SUBSTR(pesel, 6, 1) * 3 +
SUBSTR(pesel, 7, 1) * 7 + SUBSTR(pesel, 8, 1) * 9 + SUBSTR(pesel, 9, 1) * 1 +
SUBSTR(pesel, 10, 1) * 3) % 10) % 10 = SUBSTR(pesel, 11, 1) * 1;
END$$
DELIMITER ;

DELIMITER $$
CREATE TRIGGER ValidatePesel
BEFORE INSERT ON Ludzie FOR EACH ROW
BEGIN
 IF PeselDataCheck(NEW.PESEL, NEW.data_urodzenia) <> 1 OR
PeselPlecCheck(NEW.PESEL, NEW.plec) <> 1 OR PeselControlCheck(NEW.PESEL) <> 1
THEN SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'zly numer pesel';
 END IF;
END$$
DELIMITER ;

INSERT INTO Ludzie (PESEL, imie, nazwisko, data_urodzenia, plec)
VALUES('11281453517', 'imie1', 'nazwisko1', '2011-08-14', 'M'), ('09272419363', 'imie2',
'nazwisko2', '2009-07-24', 'K'), ('07290883915', 'imie3', 'nazwisko3', '2007-09-08', 'M'),
('12260625435', 'imie4', 'nazwisko4', '2012-06-06', 'M'), ('12290276115', 'imie5', 'nazwisko5',
'2012-09-03', 'M');

DELIMITER $$
CREATE PROCEDURE Generate(wiek_start INT UNSIGNED, wiek_koniec INT
UNSIGNED, n INT)
BEGIN
 DECLARE i INT DEFAULT 1;
 DECLARE data_start DATE DEFAULT CURRENT_DATE() - INTERVAL wiek_koniec
YEAR;
```

```
 DECLARE data_koniec DATE DEFAULT CURRENT_DATE() - INTERVAL wiek_start
YEAR;
 DECLARE p ENUM('K', 'M');
 DECLARE pesel CHAR(11);
 DECLARE data_ur DATE;
 DECLARE ctrl VARCHAR(1);
 DECLARE rng VARCHAR(4);
 DECLARE data_nr VARCHAR(6);
 DECLARE imie VARCHAR(30);
 DECLARE nazwisko VARCHAR(30);
 WHILE i <= n DO
  SET data_ur = TIMESTAMPADD(SECOND, FLOOR(RAND() *
TIMESTAMPDIFF(SECOND, data_start, data_koniec)), data_start);
  SET p = 1 + FLOOR(RAND() * 2);
  SET rng = 1000 + 2 * FLOOR(RAND() * 4000);
  IF p = 'M' THEN SET rng = rng + 1; END IF;
  SET data_nr = CONCAT(DATE_FORMAT(data_ur, '%y'), GetPeselMonth(data_ur),
DATE_FORMAT(data_ur, '%d'));
  SET imie = CONCAT('imie', FLOOR(RAND()*100));
  SET nazwisko = CONCAT('nazwisko', FLOOR(RAND()*100));
  SET pesel = CONCAT(data_nr, rng);
  SET ctrl = (10 - (SUBSTR(pesel, 1, 1) * 1 + SUBSTR(pesel, 2, 1) * 3 + SUBSTR(pesel, 3, 1)
* 7 + SUBSTR(pesel, 4, 1) * 9 + SUBSTR(pesel, 5, 1) * 1 + SUBSTR(pesel, 6, 1) * 3 +
SUBSTR(pesel, 7, 1) * 7 + SUBSTR(pesel, 8, 1) * 9 + SUBSTR(pesel, 9, 1) * 1 +
SUBSTR(pesel, 10, 1) * 3) % 10) % 10;
  SET pesel = CONCAT(pesel, ctrl);
  INSERT INTO Ludzie (PESEL, imie, nazwisko, data_urodzenia, plec) VALUE(pesel, imie,
nazwisko, data_ur, p);
  SET i = i + 1;
 END WHILE;
END$$
DELIMITER ;

INSERT INTO Zawody (nazwa, pensja_min, pensja_max) VALUES('polityk', '2000', '10000'),
('nauczyciel', '1500', '2800'), ('lekarz', '1700', '5000'), ('informatyk', '2500', '8000');

DELIMITER $$
CREATE PROCEDURE ZawodyPrzypisz()
BEGIN
 DECLARE done INT DEFAULT FALSE;
 DECLARE data DATE;
 DECLARE p ENUM('K', 'M');
 DECLARE id INT;
 DECLARE z INT;
 DECLARE pensja INT;
 DECLARE c CURSOR FOR SELECT lp, data_urodzenia, plec FROM Ludzie;
 DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = TRUE;
 OPEN c;
 read_loop: LOOP
```

```
  FETCH c INTO id, data, p;
  IF done THEN
   LEAVE read_loop;
  END IF;
  IF TIMESTAMPDIFF(YEAR, data, CURRENT_DATE()) >= 18 THEN
   IF TIMESTAMPDIFF(YEAR, data, CURRENT_DATE()) > 65 OR
(TIMESTAMPDIFF(YEAR, data, CURRENT_DATE()) > 60 AND p = 'K') THEN
    SELECT zawod_id INTO z FROM Zawody WHERE nazwa <> 'lekarz' ORDER BY
RAND() LIMIT 1;
   ELSE
    SELECT zawod_id INTO z FROM Zawody ORDER BY RAND() LIMIT 1;
   END IF;
   SET pensja = (SELECT pensja_min FROM Zawody WHERE zawod_id = z) + (SELECT
pensja_max - pensja_min FROM Zawody WHERE zawod_id = z) * RAND();
   INSERT INTO Pracownicy VALUE(id, z, pensja);
  END IF;
 END LOOP;
 CLOSE c;
END$$
DELIMITER ;

CALL ZawodyPrzypisz;
```

2. ```
   CREATE INDEX INameGender ON Ludzie(plec, imie);
   CREATE INDEX ISalary ON Pracownicy(pensja);

   EXPLAIN SELECT * FROM Ludzie WHERE plec = 'K' AND imie LIKE 'A%'; # używa
   indeksu INameGender

   EXPLAIN SELECT * FROM Ludzie WHERE plec = 'K'; # nie używa indeksu

   EXPLAIN SELECT * FROM Ludzie WHERE imie LIKE 'K%'; # nie używa indeksu

   EXPLAIN SELECT * FROM Ludzie INNER JOIN Pracownicy USING(lp) WHERE pensja <
   2000; # używa indeksu ISalary dla Pracownicy, PRIMARY dla Ludzie

   EXPLAIN SELECT * FROM Ludzie INNER JOIN Pracownicy USING(lp) INNER JOIN
   Zawody USING(zawod_id) WHERE plec = 'M' AND nazwa = 'informatyk' AND pensja >
   10000; # używa indeksu ISalary dla Pracownicy, PRIMARY dla Ludzie, PRIMARY dla
   Zawody

   # indeksy(Ludzie): PRIMARY, INameGender, INameGender
   # indeksy(Pracownicy): lp, zawod_id, ISalary
   # indeksy(Zawody): PRIMARY
   ```
3. ...
4. ```
   SET @zawod = 'lekarz';
   PREPARE LiczbaKobiet FROM 'SELECT COUNT(*) FROM Ludzie INNER JOIN
   Pracownicy USING(lp) INNER JOIN Zawody USING(zawod_id) WHERE plec = \'K\' AND
   nazwa = ?';
   ```

EXECUTE LiczbaKobiet USING @zawod;
DEALLOCATE PREPARE LiczbaKobiet;

5. # $ mysqldump -u root -p ludziedb -R > ludziedbdump.sql
   DROP DATABASE ludziedb;
   CREATE DATABASE ludziedb;
   # $ mysqldump -u root -p --database=ludziedb < ludziedbdump.sql
6. INTRO:
   (1) ...
   (2) SELECT department FROM employees WHERE first_name = 'Bob' AND last_name = 'Franco';
   (3) UPDATE Employees SET department = 'Sales' WHERE first_name = 'Tobi' AND last_name = 'Barnett';
   (4) ALTER TABLE Employees ADD phone VARCHAR(20);
   (5) GRANT ALL ON grant_rights TO unauthorized_user;
   (6) ...
   (7) ...
   (8) ...
   (9) SELECT * FROM user_data WHERE first_name = 'John' AND last_name = 'Smith' OR '1' = '1';
   (10) Login_Count: 123
   User_Id: 123 OR 1 = 1
   (11) Employee Name: 123
   Authentication TAN: ' OR 1=1 –
   (12) Employee Name: 123
   Authentication TAN: '; UPDATE employees SET SALARY = 100000 WHERE USERID = 37648; –
   (13) Action contains: '; DROP TABLE access_log; –
   ADVANCED:
   (14) ...
   (15) ...
   (16) Name: '; SELECT * FROM user_system_data; --
   Name: ' UNION SELECT 1, user_name, password, cookie, 'a', 'a', 1 FROM user_system_data; --
   Dave: passW0rD
   (17) ...
   (18) tom' AND '1' = '1 # register jest podatny na sql injection
   tom' AND SUBSTRING(password, 1, 1) ='a # zgadujemy hasło litera po literze (thisisasecretfortomonly)
   (19) Odpowiedzi: 4, 3, 2, 3, 4