14. Aliohjelmat

a)

```
DELIMITER //
CREATE PROCEDURE AddNewTeos(
    IN p_otsikko VARCHAR(255),
    IN p_julkaisuvuosi YEAR,
    IN p_kieli VARCHAR(50),
    IN p_tekijat TEXT
)
BEGIN     DECLARE last_teos_id INT;
    DECLARE tekija_id INT;
    DECLARE done INT DEFAULT 0;
    DECLARE tekijat_cursor CURSOR FOR
        SELECT
CAST(SUBSTRING_INDEX(SUBSTRING_INDEX(p_tekijat, ',', n.n),
',', -1) AS UNSIGNED)
        FROM (SELECT 1 AS n UNION ALL SELECT 2 UNION ALL
SELECT 3 UNION ALL SELECT 4 UNION ALL SELECT 5) n
WHERE n.n <= 1 + LENGTH(p_tekijat) -
LENGTH(REPLACE(p_tekijat, ',', ''));
    DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = 1;

    INSERT INTO Teos(otsikko, julkaisuvuosi, kieli)
    VALUES (p_otsikko, p_julkaisuvuosi, p_kieli);
    SET last_teos_id = LAST_INSERT_ID();

    OPEN tekijat_cursor;
    read_loop: LOOP         FETCH tekijat_cursor INTO
tekija_id;
        IF done THEN             LEAVE read_loop;
        END IF;
        INSERT INTO Work_Author(idTeos, idTekijä)
        VALUES (last_teos_id, tekija_id);
    END LOOP;
    CLOSE tekijat_cursor;
END;
//
DELIMITER ;
```

Esimerkki käyttö:

```
CALL AddNewTeos('Uusi Kirja', 2025, 'suomi', '10,11');
```

b)

```
DELIMITER //

CREATE PROCEDURE AddNewLainaus(
    IN p_kirja_id INT,
    IN p_asiakas_id INT,
    IN p_lainauspaiva DATE
)
BEGIN
    DECLARE kirja_tila ENUM('hyllyssä','lainassa');

    SELECT tila INTO kirja_tila FROM Kirja WHERE kirja_id =
p_kirja_id;

    IF kirja_tila = 'lainassa' THEN
        SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Kirja
on jo lainassa';
    ELSE
        INSERT INTO Lainaus(kirja_id, asiakas_id,
lainauspäivä, palautuspäivä, palautettu)
        VALUES (p_kirja_id, p_asiakas_id, p_lainauspaiva,
NULL, FALSE);

        UPDATE Kirja SET tila='lainassa' WHERE
kirja_id=p_kirja_id;
    END IF;
END;
//

DELIMITER ;
```

Esimerkki käyttö:

```
CALL AddNewLainaus(1, 1, CURDATE());
```

c)

```
DELIMITER //

CREATE PROCEDURE MyohassaKirjat()
BEGIN
    SELECT k.kirja_id, t.otsikko, a.etunimi, a.sukunimi,
l.palautuspäivä     FROM Lainaus l
    JOIN Kirja k ON l.kirja_id = k.kirja_id
    JOIN Teos t ON k.teos_id = t.teos_id
    JOIN Asiakas a ON l.asiakas_id = a.asiakas_id
    WHERE l.palautettu = FALSE
      AND l.palautuspäivä IS NOT NULL
      AND l.palautuspäivä < CURDATE();
END;
//

DELIMITER ;
```

Esimerkki käyttö:

```
1. CALL MyohassaKirjat();
```

d) ja e)

```
DELIMITER //
CREATE PROCEDURE AddArviointi(
    IN p_etunimi VARCHAR(50),
    IN p_sukunimi VARCHAR(50),
    IN p_koodi VARCHAR(20),
    IN p_arvosana INT
)
BEGIN
    DECLARE opp_id INT;
    DECLARE jakso_id INT;

    IF p_arvosana < 0 OR p_arvosana > 5 THEN
        SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT =
'Arvosanan oltava välillä 0-5';
    END IF;

    SELECT idOpiskelija INTO opp_id
    FROM Opiskelija
    WHERE Etunimi=p_etunimi AND Sukunimi=p_sukunimi;
    IF opp_id IS NULL THEN
        SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT =
'Opiskelijaa ei löytynyt';
    END IF;

    SELECT idOpintojakso INTO jakso_id
    FROM Opintojakso
    WHERE Koodi = p_koodi;
    IF jakso_id IS NULL THEN
        SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT =
'Opintojaksoa ei löytynyt';
    END IF;

    INSERT INTO Arviointi (idOpiskelija, idOpintojakso,
arvosana, paivamaara)
    VALUES (opp_id, jakso_id, p_arvosana, CURDATE());
END;
//
DELIMITER ;
```

```
DELIMITER //
CREATE PROCEDURE DeleteArviointi(
    IN p_etunimi VARCHAR(50),
    IN p_sukunimi VARCHAR(50),
    IN p_koodi VARCHAR(20)
)
BEGIN
    DECLARE opp_id INT;
    DECLARE jakso_id INT;

    -- Tarkista opiskelija
    SELECT idOpiskelija INTO opp_id
    FROM Opiskelija
    WHERE Etunimi=p_etunimi AND Sukunimi=p_sukunimi;
    IF opp_id IS NULL THEN
        SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT =
'Opiskelijaa ei löytynyt';
    END IF;

    -- Tarkista opintojakso
    SELECT idOpintojakso INTO jakso_id
    FROM Opintojakso
    WHERE Koodi=p_koodi;
    IF jakso_id IS NULL THEN
        SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT =
'Opintojaksoa ei löytynyt';
    END IF;

    -- Tarkista, että arviointi löytyy
    IF NOT EXISTS (SELECT 1 FROM Arviointi WHERE
idOpiskelija=opp_id AND idOpintojakso=jakso_id) THEN
        SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT =
'Arviointia ei löytynyt';
    END IF;

    -- Poista arviointi
    DELETE FROM Arviointi WHERE idOpiskelija=opp_id AND
idOpintojakso=jakso_id;
END;
//
DELIMITER ;
```

d) ja e) esimerkki käytöt:

```
CALL AddArviointi('Aino', 'Viljakainen', 'ICT101', 5);
```

```
CALL DeleteArviointi('Aino', 'Viljakainen', 'ICT101');
```