T2 Useita tauluja – multiple Tables

T2_Q1_

1 p

Tämä on perunatietokannan taulujen rajoitteisiin ja tilaan sekä tauluihin tehtäviin kyselyihin liittyvä monivalintakysymys. Tehtävässä on 6 alikysymystä.

This is a multiple choice question related to the constraints and the state of the potato database tables as well as queries made to the tables. The task has 6 sub-questions.

Kysymys a - Question a

Onko jokaisella perunankasvattajalla agentti?
Does every potato breeder have an agent?

Kyllä - Yes
Oikein. breeder-taulun agent_id-viiteavainsarakkeelle on määritelty NOT NULL -rajoite.
Correct. For the agent_id foreign key column of the breeder table is defined a NOT NULL constraint.

Ei - No
Väärin. Katso breeder-taulun agent_id-viiteavainsarakkeelle määritellyt rajoitteet.
Wrong. See constraints defined for the agent_id foreign key column in the breeder table.

Kysymys b - Question b

Onko jokaisella agentilla yksi tai useampi perunankasvattaja, jo(i)ta agentti edustaa?
Does every agent have one or more potato breeders that the agent represents?

Kyllä - Yes
Väärin. Vertaa agent-taulun agent_id-sarakkeessa olevia arvoja breeder-taulun agent_id-sarakkeessa oleviin arvoihin, jotka viittaavat agent-tauluun.
Wrong. Compare the values in column agent_id of the agent table with the values in the agent_id column of the breeder table that refer to the agent table.

Ei - No
Oikein. agent-taulussa on agent_id-sarakkeessa id 4, johon ei ole viittauksia breeder-taulun agent_id-sarakkeesta.
Correct. The agent table has an id 4 in the agent_id column that is not referenced from the agent_id column of the breeder table.

Kysymys c - Question c

Kuinka monta riviä on breeder- ja agent-taulujen karteesisssa tulossa eli SELECT * FROM breeder, agent; -kyselyn tulostaulussa?
How many rows are in the Cartesian product of the breeder and agent table i.e. in the result table of the query SELECT * FROM breeder, agent;?

4
Väärin. Wrong.

8
Väärin. Wrong.

16
Oikein. Correct.

20
Väärin. Wrong.

Kysymys d - Question d

breeder- ja agent-taulujen tietoja halutaan liittää kyselyssä toisiinsa ja tulostauluun halutaan mukaan kaikki breeder-taulun perunankasvattajat. Valitse sopiva vaihtoehto taulujen liittämiseksi toisiinsa:
Data from the breeder and agent tables are to be joined to each other in a query and all potato breeders of the breeder table are to be included in the result table. Select the appropriate option to join the tables to each other:

```
FROM breeder, agent
WHERE breeder.agent_id = agent.agent_id
```
tai/or
```
FROM breeder INNER JOIN agent
ON breeder.agent_id = agent.agent_id
```
Oikein. Ulkoliitosta ei tarvita, koska perunankasvattajalla on aina agentti.
Correct. No outer join is needed, because the potato breeder always has an agent.

```
FROM breeder LEFT OUTER JOIN agent
ON breeder.agent_id = agent.agent_id
```
Väärin. Ulkoliitosta ei tarvita, koska perunankasvattajalla on aina agentti.
Wrong. No outer join is needed, because the potato breeder always has an agent.

```
FROM breeder RIGHT OUTER JOIN agent
ON breeder.agent_id = agent.agent_id
```
Väärin. Ulkoliitosta ei tarvita, koska perunankasvattajalla on aina agentti.
Wrong. No outer join is needed, because the potato breeder always has an agent.

```
FROM breeder FULL OUTER JOIN agent
ON breeder.agent_id = agent.agent_id
```
Väärin. Ulkoliitosta ei tarvita, koska perunankasvattajalla on aina agentti.
Wrong. No outer join is needed, because the potato breeder always has an agent.

Kysymys e - Question e
agent- ja breeder-taulujen tietoja halutaan liittää kyselyssä toisiinsa ja tulostauluun halutaan mukaan kaikki agent-taulun agentit. Valitse sopiva vaihtoehto taulujen liittämiseksi toisiinsa:

Data from the agent and breeder tables are to be joined to each other in a query and all agents of the agent table are to be included in the result table. Select the appropriate option to join the tables to each other:

```
FROM agent, breeder
WHERE agent.agent_id = breeder.agent_id
```
tai/or
```
FROM agent INNER JOIN breeder
ON agent.agent_id = breeder.agent_id
```
Väärin. Tarvitaan sopiva ulkoliitos, koska kaikilla agenteilla ei ole perunankasvattajia, joita ne edustavat.
Wrong. A suitable outer join is required, since not all agents have potato growers they represent.

```
FROM agent LEFT OUTER JOIN breeder
ON agent.agent_id = breeder.agent_id
```
Oikein. Tarvitaan vasen ulkoliitos, koska kaikilla agenteilla ei ole perunankasvattajia ja agent-taulu on annettu ensimmäisenä (vasemmalla).
Correct. A left outer join is required because not all agents have potato growers and the agent table is given first (left).

```
FROM agent RIGHT OUTER JOIN breeder
ON agent.agent_id = breeder.agent_id
```
Wrong. Tarvitaan ulkoliitos, koska kaikilla agenteilla ei ole perunankasvattajia. Huomaa kuitenkin, että agent-taulu on annettu ensimmäisenä (vasemmalla).
Wrong. An outer join is required because not all agents have potato growers. Note, however, that the agent table is given first (left).

```
FROM agent FULL OUTER JOIN breeder
ON agent.agent_id = breeder.agent_id
```
Väärin. Täyttä ulkoliitosta ei tarvita, koska riittää, että tulostaulussa on mukana kaikki agentit. Lisäksi perunankasvattajalla on aina agentti.
Wrong. There is no need for the full outer join, as it is enough to have all agents in the result table. In addition, a potato breeder always has an agent.

Kysymys f - Question f

Kuinka monta pääavainta on breeds-taululla?
How many primary keys does the breeds table have?

1
Oikein. breeds-taululla on yksi pääavain, joka koostuu kahdesta sarakkeesta. Taululla voi olla vain yksi pääavain.
Correct. The breeds table has one primary key consisting of two columns. A table can have only one primary key.

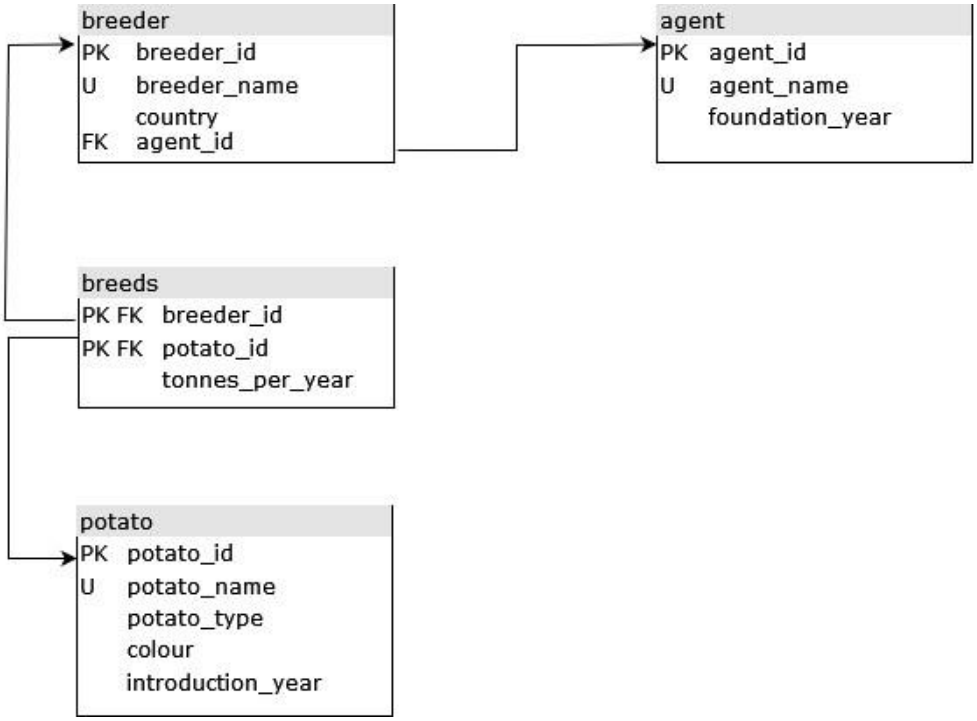
2
Väärin. Taululla voi olla vain yksi pääavain.
Wrong. A table can have only one primary key.

T2_Q2

2 p

Tarkastellaan yllä annettuja perunatietokannan taulujen luontilauseita. Piirrä perunatietokannan kaavion graafinen esitys.

Let's look at the create table statements of the potato database given above. Draw a graphical presentation of the potato database schema.



T2_Q3

1 p
Hae kaikille perunankasvattajille (breeder) seuraavat tiedot: tunnus, nimi ja agentin nimi.
Lajittele tulostaulun rivit kasvattajan tunnuksen perusteella nousevaan järjestykseen.

Retrieve the following information for all potato breeders: id, name, and agent's name.
Sort the result rows in ascending order by breeder's id.

Tulossarakkeet - Result columns:
breeder_id | breeder_name | agent_name

```
SELECT breeder_id, breeder_name, agent_name
FROM breeder, agent
WHERE breeder.agent_id = agent.agent_id
ORDER BY breeder_id;
```

```
breeder_id  breeder_name      agent_name
----------  ----------------  ------------------
1           Chips             Potatis Institutet
2           Sipsix            Perunatukku
3           Kartoffelmeister  Agrico
4           Penan pottu       Agrico
```

T2_Q4

1 p

Hae seuraavat tiedot perunankasvattajille, joiden maa (country) on Finland: tunnus, nimi ja agentin nimi.
Lajittele tulostaulun rivit kasvattajan tunnuksen perusteella nousevaan järjestykseen.

Retrieve the following information for potato breeders, whose country is Finland: id, name, and agent's name.
Sort the result rows in ascending order by breeder's id.

Tulossarakkeet - Result columns:
breeder_id | breeder_name | agent_name

```
SELECT breeder_id, breeder_name, agent_name
FROM breeder, agent
WHERE breeder.agent_id = agent.agent_id AND
      country = 'Finland'
ORDER BY breeder_id;
```

```
breeder_id  breeder_name  agent_name
----------  ------------  -----------
2           Sipsix        Perunatukku
4           Penan pottu   Agrico
```

T2_Q5

2 p

Hae seuraavat tiedot kaikille agenteille: agentin tunnus ja nimi sekä agentin edustaman perunankasvattajan tunnus ja nimi. Tulostaulussa tulee olla mukana myös ne agentit, joilla ei ole perunankasvattajia tietokannassa.
Lajittele tulosrivit agentin tunnuksen ja perunankasvattajan tunnuksen perusteella nousevaan järjestykseen.

Retrieve the following information for all agents: agent's ID and name as well as IDs and names of the potato breeders represented by the agent. The result table must also include those agents who do not have any potato breeders in the database.
Sort the result rows in ascending order by agent ID and breeder ID.

Tulossarakkeet - Result columns:
agent_id | agent_name | breeder_id | breeder_name

```
SELECT agent.agent_id,
       agent.agent_name,
       breeder.breeder_id,
       breeder.breeder_name
FROM agent
     LEFT OUTER JOIN breeder
     ON agent.agent_id = breeder.agent_id
ORDER BY agent.agent_id, breeder_id;
```

```
agent_id  agent_name          breeder_id  breeder_name
--------  ------------------  ----------  ----------------
1         Potatis Institutet  1           Chips
2         Agrico              3           Kartoffelmeister
2         Agrico              4           Penan pottu
3         Perunatukku         2           Sipsix
4         Kartoffelzentrum
```

T2_Q6

2 p

Hae perunalajikkeiden nimet ja niitä kasvattavien kasvattajien nimet.
Lajittele tulostaulun rivit perunalajikkeen nimen ja kasvattajan nimen perusteella nousevaan järjestykseen.

Retrieve the names of potato varieties and the names of the breeders breeding them.
Sort the result rows in ascending order by potato variety name and breeder name.

Tulossarakkeet - Result columns:
potato_name | breeder_name

```
SELECT potato_name, breeder_name
FROM potato, breeds, breeder
WHERE potato.potato_id = breeds.potato_id AND
      breeds.breeder_id = breeder.breeder_id
ORDER BY potato_name, breeder_name;
```

```
potato_name    breeder_name
-------------  ----------------
Bambino        Sipsix
Blue Annelise  Sipsix
Cara           Chips
Duke of York   Chips
Duke of York   Kartoffelmeister
Dunbar Rover   Chips
Dunbar Rover   Penan pottu
```

T2_Q7

2 p

Hae kaikkien perunalajikkeiden nimet ja niitä kasvattavien kasvattajien nimet. Tulostaulussa tulee olla mukana myös ne perunalajikkeet, joita mikään kasvattaja ei kasvata.
Lajittele tulostaulun rivit perunalajikkeen nimen ja kasvattajan nimen perusteella nousevaan järjestykseen.

Retrieve the names of all potato varieties and the names of the breeders breeding them. The result table must also include the potato varieties that are not bred by any breeder.
Sort the result rows in ascending order by potato variety name and breeder name.

Tulossarakkeet - Result columns:
potato_name | breeder_name

```
SELECT potato_name, breeder_name
FROM potato
     LEFT OUTER JOIN breeds
     ON potato.potato_id = breeds.potato_id
     LEFT OUTER JOIN breeder
     ON breeds.breeder_id = breeder.breeder_id
ORDER BY potato_name, breeder_name;
```

```
potato_name    breeder_name
-------------  ----------------
Bambino        Sipsix
Blue Annelise  Sipsix
Cara           Chips
Duke of York   Chips
```

```
Duke of York   Kartoffelmeister
Dunbar Rover   Chips
Dunbar Rover   Penan pottu
Moonlight
```

T2_Q8

2 p

Hae niiden perunankasvattajien nimet, jotka kasvattavat Dunbar Rover -nimistä perunalajiketta. Lajittele tulostaulun rivit perunankasvattajan nimen perusteella nousevaan järjestykseen.

Retrieve the names for those potato breeders who breed the potato variety named Dunbar Rover.
Sort the result rows in ascending order by the breeder name.

Tulossarakkeet - Result columns:
breeder_name

```
SELECT breeder_name
FROM breeder, breeds, potato
WHERE breeder.breeder_id = breeds.breeder_id AND
      breeds.potato_id = potato.potato_id AND
      potato_name = 'Dunbar Rover'
ORDER BY breeder_name;
```

```
breeder_name
------------
Chips
Penan pottu
```

T2_Q9

2 p

Hae tunnukset ja nimet niille perunalajikkeille, joita ainakin yksi perunankasvattaja kasvattaa. Kunkin lajikkeen tunnuksen ja nimen tulee esiintyä tulostaulussa vain kerran.
Lajittele tulostaulun rivit perunalajikkeen tunnuksen perusteella nousevaan järjestykseen.

Retrieve IDs and names for potato varieties that are bred at least by one breeder. The ID and name of each variety should appear only once in the result table.
Sort the result rows in ascending order by potato variety ID.

Tulossarakkeet - Result columns:
potato_id | potato_name

```
SELECT DISTINCT potato.potato_id, potato_name
FROM potato
     INNER JOIN breeds
     ON potato.potato_id = breeds.potato_id
ORDER BY potato.potato_id;
```

```
potato_id  potato_name
---------  -------------
1          Dunbar Rover
2          Bambino
3          Blue Annelise
4          Duke of York
5          Cara
```

T2_Q10

2 p

Hae tunnukset ja nimet niille perunalajikkeille, joita mikään perunankasvattaja ei kasvata.
Lajittele tulostaulun rivit perunalajikkeen tunnuksen perusteella nousevaan järjestykseen.

Retrieve IDs and names for potato varieties that are not bred by any breeder.
Sort the result rows in ascending order by potato variety id.

Tulossarakkeet - Result columns:
potato_id | potato_name

```
SELECT potato.potato_id, potato_name
FROM potato
     LEFT OUTER JOIN breeds
     ON potato.potato_id = breeds.potato_id
WHERE breeds.potato_id IS NULL
ORDER BY potato.potato_id;
```

```
potato_id  potato_name
---------  -----------
6          Moonlight
```

T2_Q11

Tämä on INSERT INTO-, UPDATE- ja DELETE-lauseisiin ja perunatietokannan taulujen rajoitteisiin ja tilaan liittyvä monivalintakysymys, jossa on 9 alikysymystä.

This is a multiple choice question related to the INSERT INTO, UPDATE and DELETE statements and the constraints and state of the tables of the potato database, with 9 sub-questions.

Suorittaako SQL-komentotulkki alla annetun operaation?
Does the SQL shell run the operation given below?

Kysymys a - Question a

```
UPDATE breeder
SET agent_id = 1
WHERE breeder_id = 3;
```

Kyllä - Yes
Oikein. Viittauksen kohde (agent-taulun agent_id-sarakkeessa arvo 1) on olemassa ja on olemassa perunankasvattaja, jonka breeder_id on 3.
Correct. The target of the reference (value 1 in the agent_id column of the agent table) exists and there is a potato breeder whose breeder_id is 3.

Ei - No
Väärin. Wrong.

Kysymys b - Question b

```
INSERT INTO breeder VALUES (3, 'Crisp', 'Sweden', NULL);
```

Kyllä - Yes
Väärin. Wrong.

Ei - No
Oikein. Sarakkeessa agent_id ei sallita NULL-arvoa (NOT NULL -määre). Lisäksi arvo 3 on jo olemassa breeder_id-sarakkeessa.
Correct. Null values are not allowed in the agent_id column (NOT NULL restriction). In addition, value 3 already exists in the breeder table.

Kysymys c - Question c

```
INSERT INTO breeder VALUES (6, 'Megamix', 'England', 8);
```

Kyllä - Yes
Väärin. Wrong.

Ei - No
Oikein. Viite-eheys rikkoutuisi, koska agent-taulussa ei ole arvoa 8 agent_id-sarakkeessa.
Correct. Referential integrity would be broken, because the value 8 does not exist in the column agent_id of the agent table.

Kysymys d - Question d

```
UPDATE potato
SET potato_id = 10
WHERE potato_id = 2;
```

Kyllä - Yes
Väärin. Wrong.

Ei - No
Oikein. Viite-eheys rikkoutuisi. potato-taulun potato_id-sarakkeessa olevaan arvoon 2 on viittauksia breeds-taulusta.
Correct. Referential integrity would be broken. The value 2 in the potato_id column of the potato table is referenced from the breeds table.

Kysymys e - Question e

UPDATE potato
SET potato_id = 12
WHERE potato_id = 6;

Kyllä - Yes
Oikein. Arvo 6 potato-taulun potato_id-sarakkeessa ei ole viittauksen kohteena breeds-taulusta ja arvoa 12 ei ole potato-taulun potato_id-sarakkeessa.

Correct. The value 6 in the potato_id column of the potato table is not referenced from the breeds table and the value 12 is not present in the potato_id column of the potato table.

Ei - No
Väärin. Wrong.

Kysymys f - Question f

UPDATE potato
SET potato_name = 'Figliolo'
WHERE potato_id = 2;

Kyllä - Yes
Oikein. Sarake, johon muutos kohdistuu, ei ole viittauksen kohteena.
Correct. The column to be updated is not a target of any reference.

Ei - No
Väärin. Wrong.

Kysymys g - Question g

```
DELETE FROM agent
WHERE agent_id = 2;
```

Kyllä - Yes
Väärin. Wrong.

Ei - No
Oikein. Viite-eheys rikkoutuisi, koska agent-taulun agent_id-sarakkeessa olevaan arvoon 2 on viittauksia breeder-taulusta.
Correct. Referential integrity would be broken, because the value 2 in the agent_id column of the agent table is referenced from the breeder table.

Kysymys h - Question h

```
INSERT INTO breeds
VALUES (3, 4, 250);
```

Kyllä - Yes
Väärin. Wrong.

Ei - No
Oikein. breeds-taulussa on jo pääavainyhdistelmä (3,4).
Correct. The breeds table has already the primary key combination (3,4).

Kysymys i - Question i

```
UPDATE breeds
SET potato_id = 4
WHERE breeder_id = 4 AND potato_id = 1;
```

Kyllä - Yes
Oikein. Päivityksen kohteena oleva rivi on olemassa breeds-taulussa ja potato-taulussa on arvo 4 potato_id-sarakkeessa. Lisäksi yhdistelmää (4,4) ei ole vielä breeds-taulussa pääavainsarakkeissa.
Correct. The row to be updated exists in the breeds table and potato table has the value 4 in the potato_id column. Furthermore, the combination (4,4) does not exist in the primary key columns of the breeds table.

Ei - No
Väärin. Wrong.

T2_Q12

2 p
Edellisen viikon harjoituksissa käytettiin bike-nimistä taulua.

Avaa ensin SQLitessa tietokanta, joka sisältää bike-taulun, tai luo bike-taulu ja lisää siihen tietoja.

Laita seuraavaksi SQLitessa viite-eheyden valvonta päälle antamalla komento PRAGMA foreign_keys = ON;

Tee luontilause bike_shop-nimiselle taululle, johon voidaan tallentaa tietoja pyöräkaupoista. Taulussa on seuraavat sarakkeet:

- Taulun pääavain on id-sarake, jossa ei sallita puuttuvia arvoja (NULL-arvoja).
- name-sarake on yksilöivä vaihtuvamittainen merkkijono ja siinä ei sallita puuttuvia arvoja.
- address on vaihtuvamittainen merkkijono

Luo taulu SQLitella. Lisää tauluun muutaman pyöräkaupan tiedot; keksi pyöräkauppojen tunnus-, nimi- ja osoitetiedot itse.

Tee luontilause sells-taululle, joka yhdistää pyörät ja pyöräkaupat toisiinsa: Yhdessä pyöräkaupassa voidaan myydä useaa eri pyörää ja samaa pyörää voidaan myydä useassa eri pyöräkaupassa. Määrittele sells-tauluun viiteavainsarakkeet, jotka viittaavat bike-tauluun ja bike_shop-tauluun sekä pääavain. Sisällytä viiteavainsarakkeiden nimiin niiden taulujen nimet, joihin viiteavainsarakkeet viittaavat.

Luo taulu SQLitella ja lisää siihen muutama rivi tietoja.

Huom. Määrittele pääavaimet, avaimet (yksilöivä, unique) ja viiteavaimet luontilauseen lopussa oppimateriaalien esimerkkien tapaan automaattisen tarkastuksen vuoksi.

In the previous week's exercises we used a table called bike.

In SQLite, first open the database containing the bike table or create the bike table and add data to it.

Next, in SQLite, turn the referencial integrity control on by entering the command PRAGMA foreign_keys = ON;

Make a creation statement for a table called bike_shop, where information about bike shops can be stored. The table has the following columns:

- The primary key of the table is the id column, which does not allow missing values (NULL values).
- The name column is a unique variable-length string and does not allow missing values.
- address is a variable-length character string

Create the table with SQLite. Insert data of a few bike shops to the table; you can make up the id, name and address data of the bike shops.

Make a creation statement for a table sells that connects bikes and bike shops: Several different bikes can be sold in one bike shop and the same bike can be sold in several different bike shops. Define foreign key columns for the sells table that refer to the bike table and bike_shop table, as well as the primary key. In the foreign key column names, include the names of the tables that the foreign key columns refer to.

Create the table with SQLite and add a few rows of data to it.

N.B. Specify the primary key, keys (unique) and foreign keys at the end of the creation statement, as in the examples in study materials, for automatic evaluation.

```
PRAGMA foreign_keys = ON;

-- Kirjoita SQL-lauseesi alle:
-- Write your SQL statements below:

CREATE TABLE bike_shop (
    id INT,
    name VARCHAR(30) NOT NULL,
    address VARCHAR(50),
    PRIMARY KEY (id),
    UNIQUE (name)
);

INSERT INTO bike_shop
VALUES (1, 'The Bike Shop', 'Kauppakatu 2');

INSERT INTO bike_shop
VALUES (2, 'BTech', 'Pyöräkeskuksentie 7');


 CREATE TABLE sells (
    bike_id INT NOT NULL,
    bike_shop_id INT NOT NULL,
    PRIMARY KEY (bike_id, bike_shop_id),
    FOREIGN KEY (bike_id) REFERENCES bike,
    FOREIGN KEY (bike_shop_id) REFERENCES bike_shop
);

INSERT INTO sells
```

```
VALUES (1, 1);

INSERT INTO sells
VALUES (2, 1);

INSERT INTO sells
VALUES (4, 1);

INSERT INTO sells
VALUES (2, 2);

INSERT INTO sells
VALUES (3, 2);
```