Modul 6 -
Fișă de lucru

Problema1. Să se creeze o bază de date "student" cu tabela „Datepers" ce să conțină câmpurile (idstudent, nume, prenume, an, grupa).

Rezolvare

```
CREATE DATABASE IF NOT EXISTS student;

USE student;

  CREATE TABLE IF NOT EXISTS `Datepers` (
 `idstudent` int(11) NOT NULL AUTO_INCREMENT,
 `nume` varchar(100) COLLATE utf8_unicode_ci NOT NULL,
 `prenume` varchar(150) COLLATE utf8_unicode_ci NOT NULL,
 `an` INT(3) NOT NULL,
 `grupa` INT(3) NOT NULL,
 PRIMARY KEY (`idstudent`)
 ) ;
```

Problema2. Să se afișeze structura tabelei Datepers.

Rezolvare

```
describe Datepers;
```

Problema3. Să se insereze 3 înregistrări în tabela Datepers

Rezolvare

```
INSERT INTO Datepers (idstudent,nume, prenume, an, grupa) VALUES
(NULL,'Thomas', 'Carter', 1, 2);
INSERT INTO Datepers (idstudent,nume, prenume, an, grupa) VALUES
(NULL, 'Rachel', 'Rodriguez', 2, 3);
INSERT INTO Datepers (idstudent,nume, prenume, an, grupa) VALUES
(NULL, 'Harry', 'Leraderr', 1, 1);
```

Problema4. Să se afișeze înregistrările din tabela Datepers

Rezolvare

```
Select * from datepers;
```

Problema 5. Să se creeze o bază de date “magazin” cu tabelele Produse(produs_id, produs_nume, produs_pret, produs_img, produs_categ, produs_descriere, produs_desccompl, produs_stare,produs_oferta,produs_noutati)

Clienti(client_id,client_username,client_pass,client_email, client_str, client_oras, client_tara, client_codpost,client_nrcard,client_tipcard,client_dataexp,acceptareemail,clien _nume,client_nrinregRC,cod_fiscal)

Ordin(ordin_id,ordin_prodID, ordin_calit, ordin_client_id, ordin_dataintr, ordin_stare, ordin_shipdate)

Situatievizita(id,numepagviz,platforma,referrer,time, date, host)

Parola(userid, pass)

Cos(cos_id,cos_clientID,cos_produsID,cos_cantitate)

Rezolvare

```
CREATE DATABASE IF NOT EXISTS magazin;

USE magazin;

  CREATE TABLE IF NOT EXISTS ` Clienti ` (
 `client_id` int(11) NOT NULL AUTO_INCREMENT,
 `client_username` varchar(100) COLLATE utf8_unicode_ci NOT NULL,
 `client_pass` varchar(150) COLLATE utf8_unicode_ci NOT NULL,
 `client_email` varchar(150) COLLATE utf8_unicode_ci NOT NULL,
 `client_str` varchar(150) COLLATE utf8_unicode_ci NOT NULL,
 `client_oras` varchar(150) COLLATE utf8_unicode_ci NOT NULL,
 `client_tara` varchar(150) COLLATE utf8_unicode_ci NOT NULL,
 `client_codpost` varchar(150) COLLATE utf8_unicode_ci NOT NULL,
 `client_nrcard` INT(100) NOT NULL,
 `client_tipcard` varchar(150) COLLATE utf8_unicode_ci NOT NULL,
 `client_dataexp` DATETIME NOT NULL,
 ` acceptareemail ` INT(3) NOT NULL,
 ` client_nrinregRC ` INT(100) NOT NULL,
 `client_nume` varchar(150) COLLATE utf8_unicode_ci NOT NULL,
 ` cod_fiscal ` INT(100) NOT NULL,
 PRIMARY KEY (`client_id `)
 ) ;
```

```
CREATE TABLE IF NOT EXISTS `Produse` (
`produs_id` int(11) NOT NULL AUTO_INCREMENT,
`produs_nume` varchar(100) COLLATE utf8_unicode_ci NOT NULL,
`produs_pret` DECIMAL(13,2) NOT NULL,
`produs_img` varchar(150) COLLATE utf8_unicode_ci NOT NULL,
`produs_categ` varchar(150) COLLATE utf8_unicode_ci NOT NULL,
`produs_descriere` varchar(250) COLLATE utf8_unicode_ci NOT NULL,
`produs_desccompl`  varchar(1250)  COLLATE  utf8_unicode_ci  NOT
NULL,
`produs_stare` varchar(150) COLLATE utf8_unicode_ci NOT NULL,
`produs_oferta` INT(2) NOT NULL,
`produs_noutati` INT(2) NOT NULL,
PRIMARY KEY (`produs_id`)
) ;

CREATE TABLE IF NOT EXISTS `Ordin` (
`ordin_id` int(11) NOT NULL AUTO_INCREMENT,
`ordin_prodID` int(11) NOT NULL,
`ordin_cantit` int(11) NOT NULL,
`ordin_client_id` int(11) NOT NULL,
`ordin_dataintr` DATETIME NOT NULL,
`ordin_stare` varchar(150) COLLATE utf8_unicode_ci NOT NULL,
`ordin_shipdate` DATETIME NOT NULL,
PRIMARY KEY (`ordin_id`)
) ;

CREATE TABLE IF NOT EXISTS `Cos` (
`cos_id` int(11) NOT NULL AUTO_INCREMENT,
`cos_clientID ` int(11) NOT NULL,
`cos_produsID` int(11) NOT NULL,
`cos_cantitate` int(11) NOT NULL,
PRIMARY KEY (`cos_id`)
) ;


CREATE TABLE IF NOT EXISTS `Situatievizita` (
`id` int(11) NOT NULL AUTO_INCREMENT,
`numepagviz` varchar(150) COLLATE utf8_unicode_ci NOT NULL,
`platforma ` varchar(150) COLLATE utf8_unicode_ci NOT NULL,
`referrer ` varchar(150) COLLATE utf8_unicode_ci NOT NULL,
`time` TIMESTAMP NOT NULL,
`date` DATETIME NOT NULL,
`host` varchar(150) COLLATE utf8_unicode_ci NOT NULL,
PRIMARY KEY (`id`)
) ;

CREATE TABLE IF NOT EXISTS `Parola` (
`userid` int(11) NOT NULL AUTO_INCREMENT,
`pass` varchar(350) COLLATE utf8_unicode_ci NOT NULL,
PRIMARY KEY (`userid`) ) ;
```

Problema 6. Să se populeze fiecare tabelă din bază de date "magazin" cu câte 5 interogări.

Problema 7. Să se creeze o bază de date ce conţine informaţii despre animalele de companie tratate în cadrul unui cabinet veterinar. Fiecare animal se identifică prin nume, rasă, gen, data naşterii şi eventual data morţii, aparţinător și opțional despre părinți (dacă se află și aceștia în evidența aceleiași unități). Fișa medicală a animalului conține informații legate de fiecare consultație, unde se va preciza data la care a fost efectuată, medicul care a realizat examinarea, diagnosticul pus și tratamentul aplicat. [Rosu,2014]

Tabele bazei de date cabinet_veterinar sunt: [Rosu,2014]

1. breed (id, name , description)
1. owner (id, first_name,last_name , email,   phone_number, bank_account);
2. animal (id, name, breed_id, gender , birth_date, death_date, owner_id, father_id ,   mother_id);
3. doctor (id , first_name ,last_name, title, speciality,code ,email, phone_number
4. diagnosis (id ,name, description, severity,cure)
5. medical_record(animal_id , doctor_id, diagnosis_id, date);

Tabele bazei de date catalog sunt: [Rosu,2014]

1. title (id, name, description);
2. speciality (id, name,description);
3. doctor (id ,first_name, last_name, title_id , speciality_id , code, email, phone_number);

```
CREATE DATABASE IF NOT EXISTS cabinet_veterinar.;

USE cabinet_veterinar;
```

```
CREATE DATABASE IF NOT EXISTS cabinet_veterinar;

USE cabinet_veterinar;

CREATE TABLE IF NOT EXISTS breed (
    id            INT(10) UNSIGNED AUTO_INCREMENT NOT NULL,
    name            VARCHAR(50) NOT NULL,
    description     VARCHAR(1000),
    KEY (id)
);
ALTER TABLE breed ADD CONSTRAINT pk_breed_id PRIMARY KEY (id);

CREATE TABLE IF NOT EXISTS owner (
    id            INT(10) UNSIGNED AUTO_INCREMENT NOT NULL,
    first_name      VARCHAR(50) NOT NULL,
    last_name       VARCHAR(50) NOT NULL,
    email           VARCHAR(50),
    phone_number    INT(10) NOT NULL,
    bank_account    VARCHAR(50) NOT NULL,
    KEY (id)
);
ALTER TABLE owner ADD CONSTRAINT pk_owner_id PRIMARY KEY (id);
ALTER TABLE owner ADD CONSTRAINT owner_email_format CHECK (email
LIKE '%@%.%');

CREATE TABLE IF NOT EXISTS animal (
    id            INT(10) UNSIGNED AUTO_INCREMENT NOT NULL,
    name            VARCHAR(50) NOT NULL,
    breed_id        INT(10) UNSIGNED,
    gender          CHAR(1) DEFAULT 'M' NOT NULL,
    birth_date      DATETIME NOT NULL,
    death_date      DATETIME,
    owner_id        INT(10) UNSIGNED,
    father_id       INT(10) UNSIGNED,
    mother_id       INT(10) UNSIGNED,
    KEY (id)
);
ALTER TABLE animal ADD CONSTRAINT pk_animal_id PRIMARY KEY (id);
ALTER TABLE animal ADD CONSTRAINT fk_animal_breed_id FOREIGN
KEY(breed_id) REFERENCES breed(id) ON UPDATE CASCADE ON DELETE
SET NULL;
```

```
ALTER TABLE animal ADD CONSTRAINT fk_animal_ownder_id FOREIGN
KEY(owner_id) REFERENCES owner(id) ON UPDATE CASCADE ON DELETE
SET NULL;
ALTER TABLE animal ADD CONSTRAINT fk_animal_father_id FOREIGN
KEY(father_id) REFERENCES animal(id) ON UPDATE CASCADE ON DELETE
SET NULL;
ALTER TABLE animal ADD CONSTRAINT fk_animal_mother_id FOREIGN
KEY(mother_id) REFERENCES animal(id) ON UPDATE CASCADE ON DELETE
SET NULL;
ALTER TABLE animal ADD CONSTRAINT chk_animal_sex_possible_values
CHECK (gender in ('M', 'F'));

CREATE TABLE IF NOT EXISTS doctor (
   id              INT(10) UNSIGNED AUTO_INCREMENT NOT NULL,
   first_name      VARCHAR(50) NOT NULL,
   last_name       VARCHAR(50) NOT NULL,
   title           VARCHAR(20) NOT NULL,
   speciality      VARCHAR(20),
   code            VARCHAR(20) NOT NULL,
   email           VARCHAR(50),
   phone_number    INT(10) NOT NULL,
   KEY (id)
);
ALTER TABLE doctor ADD CONSTRAINT pk_doctor_id PRIMARY KEY (id);
ALTER TABLE doctor ADD CONSTRAINT chk_doctor_email_format CHECK
(email LIKE '%@%.%');

CREATE TABLE IF NOT EXISTS diagnosis (
   id              INT(10) UNSIGNED AUTO_INCREMENT NOT NULL,
   name            VARCHAR(50) NOT NULL,
   description     VARCHAR(1000) NOT NULL,
   severity        VARCHAR(20) NOT NULL,
   cure            VARCHAR(1000) NOT NULL,
   KEY (id)
);
ALTER TABLE diagnosis ADD CONSTRAINT pk_diagnosis_id PRIMARY KEY
(id);

CREATE TABLE IF NOT EXISTS medical_record (
   id              INT(10) UNSIGNED AUTO_INCREMENT NOT NULL,
   animal_id       INT(10) UNSIGNED,
   doctor_id       INT(10) UNSIGNED,
   diagnosis_id    INT(10) UNSIGNED,
   date           DATETIME NOT NULL,
   treatment       VARCHAR(1000),
   KEY (id)
);
ALTER TABLE medical_record ADD CONSTRAINT pk_medical_record_id
PRIMARY KEY (id);
ALTER TABLE medical_record ADD CONSTRAINT
fk_medical_record_animal_id FOREIGN KEY(animal_id) REFERENCES
animal(id) ON UPDATE CASCADE ON DELETE SET NULL;
ALTER TABLE medical_record ADD CONSTRAINT
fk_medical_record_doctor_id FOREIGN KEY(doctor_id) REFERENCES
doctor(id) ON UPDATE CASCADE ON DELETE SET NULL;
ALTER TABLE medical_record ADD CONSTRAINT
fk_medicak_record_diagnosis_id FOREIGN KEY(diagnosis_id) REFERENCES
diagnosis(id) ON UPDATE CASCADE ON DELETE SET NULL;
```

II. Creare baza de date

```

CREATE DATABASE IF NOT EXISTS catalog;

USE catalog;

CREATE TABLE IF NOT EXISTS  title (
   id             INT(10) UNSIGNED AUTO_INCREMENT NOT NULL,
   name           VARCHAR(50) NOT NULL,
   description    VARCHAR(1000),
      KEY(id)
);
ALTER TABLE title ADD CONSTRAINT pk_title_id PRIMARY KEY (id);

CREATE TABLE IF NOT EXISTS  speciality (
   id             INT(10) UNSIGNED AUTO_INCREMENT NOT NULL,
   name           VARCHAR(50) NOT NULL,
   description    VARCHAR(1000),
      KEY(id)
);
ALTER TABLE speciality ADD CONSTRAINT pk_speciality_id PRIMARY KEY
(id);

CREATE TABLE IF NOT EXISTS  doctor (
    id              INT(10) UNSIGNED AUTO_INCREMENT NOT NULL,
    first_name      VARCHAR(50) NOT NULL,
        last_name       VARCHAR(50) NOT NULL,
    title_id        INT(10) UNSIGNED,
    speciality_id   INT(10) UNSIGNED,
    code            VARCHAR(20) NOT NULL,
    email           VARCHAR(50),
    phone_number    INT(10) NOT NULL,
        KEY (id)
);
ALTER TABLE doctor ADD CONSTRAINT pk_doctor_id PRIMARY KEY (id);
ALTER TABLE doctor ADD CONSTRAINT fk_doctor_title_id FOREIGN KEY
(title_id) REFERENCES title (id) ON UPDATE CASCADE ON DELETE SET
NULL;
ALTER TABLE doctor ADD CONSTRAINT fk_doctor_speciality_id FOREIGN
KEY (speciality_id) REFERENCES speciality (id) ON UPDATE CASCADE ON
DELETE SET NULL;


INSERT INTO title(name, description) VALUES
('fellow', '-'),
('resident', '-'),
('attending', '-'),
('professor', '-');

INSERT INTO speciality(name, description) VALUES
('surgery', '-'),
('general care', '-'),
('radiology', '-'),
('stomatology', '-');

INSERT INTO doctor(first_name, last_name, title_id, speciality_id, code,
email, phone_number) VALUES
('Thomas', 'CARTER', 1, 2, 'AAA111', 'thomas.carter@google.com', 111222),
('Rachel', 'RODRIGUEZ', 3, 1, 'BBB222', 'rachel.rodrigues@hotmail.com',
333444),
('Harry', 'LEWIS', 2, 4, 'CCC333', 'harry.lewis@live.com', 555666),
('Richard', 'ALLEN', 3, 2, 'DDD444', 'richard.allen@space.com', 777888),
('Jessica', 'WHITE', 4, 1, 'EEE555', 'jessicawhite@aim.com', 999000);
```

Fisierul animal.txt

```
Rocky       1     M     01.01.2010 4     \N     \N
Cleopatra   5     F     31.12.2014 3     \N     \N
Lola   2    F     15.06.2012 2     \N     \N
Hutch 4     M     10.03.2011 5     \N     \N
Logan       3     M     20.09.2013 1     \N     \N
Koby  1     M     05.06.2012 4     1      \N
Ivory 2     F     25.06.2014 2     \N     3
```

```
LOAD DATA LOCAL INFILE 'C:\\animals.txt'
                    INTO TABLE animal
                    FIELDS TERMINATED BY '\t' ENCLOSED BY '' ESCAPED BY
                    '\\' LINES TERMINATED BY '\r\n' STARTING BY ''
                    (name, breed_id, gender, birth_date, owner_id, father_id,
                    mother_id);
```

Exemple de populare a tabelelor definite anterior folosind instrucțiuni de tip INSERT ar putea fi:

```
USE cabinet_veterinar;

INSERT LOW_PRIORITY INTO breed (name, description) VALUES
('dog', 'The domestic dog (Canis lupus familiaris, or Canis familiaris) is a
member of the Canidae family of the mammalian order Carnivora. The term
\"domestic dog\" is generally used for both domesticated and feral varieties.
The dog was the first domesticated animal and has been the most widely
kept working, hunting, and pet animal in human history. The word \"dog\"
can also refer to the male of a canine species, as opposed to the word
\"bitch\" which refers to the female of the species.'),
('cat','The domestic cat (Felis catus or Felis silvestris catus) is a small,
usually furry, domesticated, and carnivorous mammal. It is often called a
housecat when kept as an indoor pet, or simply a cat when there is no need
to distinguish it from other felids and felines. Cats are often valued by
humans for companionship, and their ability to hunt vermin and household
pests.'),
('guinea pig', NULL),
('horse', 'The horse (Equus ferus caballus) is one of two extant subspecies
of Equus ferus. It is an odd-toed ungulate mammal belonging to the
taxonomic family Equidae. The horse has evolved over the past 45 to 55
million years from a small multi-toed creature into the large, single-toed
animal of today. Humans began to domesticate horses around 4000 BC, and
their domestication is believed to have been widespread by 3000 BC. Horses
in the subspecies caballus are domesticated, although some domesticated
populations live in the wild as feral horses. These feral populations are not
true wild horses, as this term is used to describe horses that have never
been domesticated, such as the endangered Przewalski\'s horse, a separate
subspecies, and the only remaining true wild horse. There is an extensive,
specialized vocabulary used to describe equine-related concepts, covering
everything from anatomy to life stages, size, colors, markings, breeds,
locomotion, and behavior.'),
('gold fish', NULL);

INSERT IGNORE INTO owner
  SET first_name='Sarah', last_name='LEE',
email='sarah.lee@lavabit.com', phone_number='123456789',
bank_account='US01AABB0000000001';
INSERT IGNORE INTO owner
  SET first_name='William', last_name='LOPEZ',
email='william.lopez@aim.com', phone_number='456789123',
bank_account='UK02CCDD0000000002';
INSERT IGNORE INTO owner
  SET first_name='Samuel', last_name='JONES',
email='samuel.jones@fastmail.com', phone_number='789123456',
bank_account='DE03EEFF0000000003';
INSERT IGNORE INTO owner
  SET first_name='William', last_name='THOMPSON',
email='william.thompson@lmyway.com', phone_number='123789456',
bank_account='FR04GGHH0000000004';
INSERT IGNORE INTO owner
  SET first_name='Richard', last_name='MARTIN',
email='richard.martin@myspace.com', phone_number='789456123',
bank_account='IT05IIJJ0000000005';

INSERT HIGH_PRIORITY INTO doctor(first_name, last_name, title,
speciality, code, email, phone_number)
SELECT d.first_name, d.last_name, t.name, s.name, d.code, d.email,
d.phone_number
FROM physician_catalog.doctor d, physician_catalog.title t,
physician_catalog.speciality s WHERE t.id = d.title_id AND s.id =
d.speciality_id;

INSERT INTO diagnosis VALUES
```

```
(NULL, 'giardiasis', 'Giardiasis is a protozoal, parasitic, gastrointestinal zoonotic disease in humans and domestic and wild animals.', 'medium', 'nitroimidazole derivatives, benzimidazole compounds or acridine dyes'),
(NULL, 'rabies', 'Rabies is a severely fatal, viral, neurological disease of mammals.', 'high', 'There is no treatment once the clinical signs appear.'),
(NULL, 'dermatophytosis', 'Dermatophytosis is a fungal skin disease that commonly affects humans as well as wild and domestic animals', 'low', 'Dermatophyte infections are treated with a variety of topical and oral antifungal drugs.'),
(NULL, 'mycobacteriosis', 'Mycobacteriosis is a bacterial, systemic, granulomatous skin disease that occurs in aquarium and culture food fish and can affect humans.', 'low', ' Antibiotic therapy may be warranted to prevent progression to deep infection.'),
(NULL, 'malignant catarrhal fever', 'Malignant catarrhal fever (MCF) is a serious, often fatal, viral disease affecting cattle, bison, deer, moose, exotic ruminants, and pigs.', 'severe', 'There is no cure discovered so far');
```

Exemple de actualizare a tabelelor definite anterior folosind instrucțiuni de tip UPDATE ar putea fi: Explicati efectul comenzilor?

```
UPDATE LOW_PRIORITY animal SET birth_date=CURRENT_DATE
WHERE YEAR(CURRENT_TIMESTAMP) - YEAR(birth_date) - (MONTH(CURRENT_TIMESTAMP) < MONTH(birth_date) OR (MONTH(CURRENT_TIMESTAMP) = MONTH(birth_date) AND DAY(CURRENT_TIMESTAMP) < DAY(birth_date))) < 0
ORDER BY birth_date DESC;

UPDATE medical_record mr, diagnosis d
SET mr.treatment = d.cure
WHERE mr.diagnosis_id = d.id;
```

Exemple de ștergere a unor înregistrări din cadrul tabelelor definite anterior folosind instrucțiuni de tip DELETE ar putea fi:

```
DELETE QUICK breed, animal
FROM breed, animal
WHERE animal.breed_id = breed.id AND breed.description IS NULL;

DELETE LOW_PRIORITY IGNORE FROM animal, owner
USING animal INNER JOIN owner ON animal.owner_id=owner.id
WHERE animal.death_date IS NOT NULL;
```

În situația în care se dorește afișarea listei de animale pentru care se afișează informații precum denumirea, rasa, sexul, vârsta, numele și prenumele aparținătorului, numele părinților (dacă sunt disponibile), denumirile bolilor de care au suferit, numele și prenumele medicilor care i-au tratat, limitând rezultatele la înregistrările valide (animale cu data nașterii în trecut), s-ar putea folosi următoarea interogare:

```
SELECT a.name AS name,
      b.name AS breed,
      a.gender AS gender,
      YEAR(CURRENT_TIMESTAMP) - YEAR(a.birth_date) -
(MONTH(CURRENT_TIMESTAMP) < MONTH(a.birth_date) OR
(MONTH(CURRENT_TIMESTAMP) = MONTH(a.birth_date) AND
DAY(CURRENT_TIMESTAMP) < DAY(a.birth_date))) AS age,
      CONCAT(o.first_name, ' ' , o.last_name) AS owner,
      COALESCE((SELECT name FROM animal where
id=COALESCE(a.father_id,'0')),'-') AS father,
      COALESCE((SELECT name FROM animal where
id=COALESCE(a.mother_id,'0')),'-') AS mother,
      (SELECT GROUP_CONCAT(DISTINCT d.name) FROM diagnosis d,
medical_record mr WHERE d.id = mr.diagnosis_id AND mr.animal_id=a.id)
AS diseases,
      (SELECT GROUP_CONCAT(DISTINCT CONCAT(d.first_name,' ',
d.last_name)) FROM doctor d, medical_record mr WHERE d.id =
mr.doctor_id AND mr.animal_id=a.id) AS physicians
FROM animal a, breed b, owner o
WHERE b.id = a.breed_id AND o.id = a.owner_id
HAVING age > 0;
```

Cross-Join

```
SELECT * FROM animal CROSS JOIN owner;
```

inner-join

```
SELECT * FROM animal INNER JOIN owner ON animal.owner_id = owner.id
```

outer-join, care include înregistrările comune unei laturi a legăturii, completând câmpurile care nu au corespondent pe cealaltă latură cu NULL; în acest caz, se pot folosi cuvintele cheie LEFT | RIGHT JOIN, specificând partea relaţiei ce va include toate rezultatele.

left
SELECT * FROM animal LEFT OUTER JOIN owner ON animal.owner_id = owner.id;
right
SELECT * FROM animal RIGHT OUTER JOIN owner ON animal.owner_id = owner.id;
self-join, care implică duplicarea tabelei prin utilizarea de alias-uri;
SELECT * FROM animal a1 JOIN animal a2 ON a1.father_id = a2.id OR a1.mother_id = a2.id;
unions care presupune adăugarea tuturor înregistrărilor din tabele pentru a determina suma compozită a acestora (numărul de atribute întoarse și denumirea acestora trebuie să fie aceeași pentru a se permite realizarea acestei operații)
SELECT first_name, last_name, email, phone_number, 'Pet Owner' AS role FROM owner
UNION
SELECT first_name, last_name, email, phone_number, 'Physician' AS role FROM doctor;

Exemplu. Determinarea animalului de companie cu vârsta cea mai mare aflat în evidența clinicii veterinare poate fi realizată prin instrucțiunea:

```
SELECT a.name AS name,
      YEAR(CURRENT_TIMESTAMP) - YEAR(a.birth_date) -
(MONTH(CURRENT_TIMESTAMP) < MONTH(a.birth_date) OR
(MONTH(CURRENT_TIMESTAMP) = MONTH(a.birth_date) AND
DAY(CURRENT_TIMESTAMP) < DAY(a.birth_date))) AS age
FROM animal a
WHERE birth_date = (SELECT MIN(birth_date) FROM animal);
```

Exemplu. Determinarea animalului de companie pentru care au existat consultații în cadrul clinicii veterinare poate fi realizată si prin instrucțiunea:

```
SELECT a.name AS name,
      b.name AS breed
FROM animal a INNER JOIN breed b ON a.breed_id=b.id
WHERE EXISTS (SELECT * FROM medical_record mr WHERE mr.animal_id
= a.id);
```

Exemplu. Determinarea speciei cu cea mai mare vârstă medie dintre cele aflate în evidența clinicii veterinare poate fi realizată prin instrucțiunea:

```
SELECT b.name,
       MAX(average_age)
FROM (SELECT a.breed_id,
           AVG(YEAR(CURRENT_TIMESTAMP) - YEAR(a.birth_date) -
(MONTH(CURRENT_TIMESTAMP) < MONTH(a.birth_date) OR
(MONTH(CURRENT_TIMESTAMP) = MONTH(a.birth_date) AND
DAY(CURRENT_TIMESTAMP) < DAY(a.birth_date)))) AS average_age
      FROM animal a
      GROUP BY a.breed_id) statistics, breed b
WHERE b.id = statistics.breed_id;
```

Exemplu. Determinarea numărului de consultații de care a beneficiat fiecare animal de companie în parte în cadrul clinicii veterinare poate fi realizat prin instrucțiunea:

```
SELECT a.name AS name,
       (SELECT COUNT(*) FROM medical_record mr WHERE mr.animal_id =
a.id) AS number_of_examinations
FROM animal a;
```

Bibliografie

1. *Gheorghe SABĂU, Vasile AVRAM, Ramona BOLOGA, Mihaela MUNTEAN, Marian DÂRDALĂ, Răzvan BOLOGA* – Baze de Date*, Editura Matrix Rom, Bucureşti, 2008*
2. *Dorin CÂRSTOIU* – Baze de Date*, Editura Matrix Rom, Bucureşti, 2009*
3. *Manole VELICANU, Ion LUNGU, Iuliana BOTHA, Adela BÂRA, Anda VELICANU, Emanuil*
4. *REDNIC* – Sisteme de Baze de Date Evoluate*, Editura ASE, Bucureşti, 2009*
5. *Vikram VASWANI* – MySQL. Utilizarea şi administrarea bazelor de date MySQL*, traducere de Cristian Alexe Dumitrescu, Editura Rosetti Educational, Bucureşti, 2010*
6. Carlos CORONEL, Steven MORRIS, Peter ROB – *Database Systems. Design, Implementation and* Management*, 9th Edition, Course Technology, Cengage Learning, Boston, 2011*
7. Ramez ELMASRI, Shamkant NAVATHE – *Fundamentals of Database Systems*, 6th Edition, Addison-Wesley, 2011
8. [Rosu,2014]Andrei Rosu Cojocaru, Aplicații Integrate pentru Întreprinderi 2014, http://aipi2014.andreirosucojocaru.ro/laboratoare/laborator01