Olivia Schultheis



The purpose of this database is to serve a veterinary clinic called “Priority Paws” by storing information related to veterinarians who provide care, their area of expertise, the pets, the pet owners, the treatments each animal receives, the details of the pet’s appointments, and specifics of the pet insurance if the pet owner has purchased it.

/\* The following code creates the tables **EXPERTISE, VETERINARIAN, SPECIES, TREATMENT, INSURANCE, PETOWNER, PATIENT, and APPOINTMENT**, and fills each table with records. \*/

CREATE TABLE EXPERTISE (

EXPERTISE\_ID int NOT NULL UNIQUE,

AREA\_OF\_EXPERTISE varchar(25) NOT NULL,

PRIMARY KEY(EXPERTISE\_ID)

);

CREATE TABLE VETERINARIAN (

VET\_ID int NOT NULL UNIQUE,

VET\_FNAME varchar(20) NOT NULL,

VET\_LNAME varchar(30) NOT NULL,

STATE\_LICENSE char(2) NOT NULL,

PHD\_SCHOOL varchar(100),

YRS\_PRACTICE int NOT NULL,

EXPERTISE\_ID int NOT NULL,

PRIMARY KEY(VET\_ID),

FOREIGN KEY(EXPERTISE\_ID) REFERENCES EXPERTISE(EXPERTISE\_ID) ON UPDATE CASCADE

);

CREATE TABLE SPECIES (

SPECIES\_ID int NOT NULL UNIQUE,

COMMON\_NAME varchar(25) NOT NULL,

LIFESPAN\_LOWER\_YEARS NUMERIC(3,1) NOT NULL,

LIFESPAN\_UPPER\_YEARS NUMERIC(3,1) NOT NULL,

PRIMARY KEY(SPECIES\_ID)

);

CREATE TABLE TREATMENT (

TREATMENT\_ID int NOT NULL UNIQUE,

TREATMENT\_NAME varchar(40) NOT NULL,

PRIMARY KEY(TREATMENT\_ID)

);

CREATE TABLE INSURANCE (

POLICY\_NO int NOT NULL UNIQUE,

POLICY\_HOLDER\_FNAME varchar(20) NOT NULL,

POLICY\_HOLDER\_LNAME varchar(30) NOT NULL,

INSURANCE\_CO varchar(50) NOT NULL,

DEDUCTIBLE\_DOLLARS int NOT NULL,

PRIMARY KEY(POLICY\_NO)

);

CREATE TABLE PETOWNER (

PET\_OWNER\_ID int NOT NULL,

OWNER\_FNAME varchar(20) NOT NULL,

OWNER\_LNAME varchar(30) NOT NULL,

PHONE\_NUMBER char(12),

PRIMARY KEY(PET\_OWNER\_ID)

);

CREATE TABLE PATIENT (

PATIENT\_ID int NOT NULL UNIQUE,

PATIENT\_NAME varchar(20) NOT NULL,

DIAGNOSIS varchar(40),

BIRTHDATE DATE NOT NULL,

SPECIES\_ID int NOT NULL,

VET\_ID int NOT NULL,

POLICY\_NO int,

PET\_OWNER\_ID int,

PRIMARY KEY(PATIENT\_ID),

FOREIGN KEY(SPECIES\_ID) REFERENCES SPECIES(SPECIES\_ID) ON UPDATE CASCADE,

FOREIGN KEY(VET\_ID) REFERENCES VETERINARIAN(VET\_ID),

FOREIGN KEY(POLICY\_NO) REFERENCES INSURANCE(POLICY\_NO),

FOREIGN KEY(PET\_OWNER\_ID) REFERENCES PETOWNER(PET\_OWNER\_ID) ON UPDATE CASCADE

);

CREATE TABLE APPOINTMENT (

APT\_ID int NOT NULL UNIQUE,

PATIENT\_ID int NOT NULL,

APT\_DATE DATE NOT NULL,

BEG\_TIME varchar(8) NOT NULL,

END\_TIME varchar(8) NOT NULL,

TREATMENT\_ID int,

PRIMARY KEY(APT\_ID),

FOREIGN KEY(PATIENT\_ID) REFERENCES PATIENT(PATIENT\_ID),

FOREIGN KEY(TREATMENT\_ID) REFERENCES TREATMENT(TREATMENT\_ID)

);

/\* Add records to Expertise Table \*/

INSERT INTO EXPERTISE VALUES(100, 'Oncology');

INSERT INTO EXPERTISE VALUES(101, 'Cardiology');

INSERT INTO EXPERTISE VALUES(102, 'Immunology');

INSERT INTO EXPERTISE VALUES(103, 'Pulmonology');

INSERT INTO EXPERTISE VALUES(104, 'Ophthalmology');

INSERT INTO EXPERTISE VALUES(105, 'Dermatology');

INSERT INTO EXPERTISE VALUES(106, 'Dentistry');

INSERT INTO EXPERTISE VALUES(107, 'Orthopedics');

INSERT INTO EXPERTISE VALUES(108, 'Podiatry');

INSERT INTO EXPERTISE VALUES(109, 'Neurology');

INSERT INTO EXPERTISE VALUES(110, 'Gastroenterology');

INSERT INTO EXPERTISE VALUES(111, 'Nephrology');

INSERT INTO EXPERTISE VALUES(112, 'Urology');

INSERT INTO EXPERTISE VALUES(113, 'Parasitology');

INSERT INTO EXPERTISE VALUES(114, 'Audiology');

/\* Add records to Veterinarian Table \*/

INSERT INTO VETERINARIAN VALUES(43890, 'Oliver', 'Fischer', 'GA', 'University of Georgia', 8, 105);

INSERT INTO VETERINARIAN VALUES(69504, 'Francis', 'Karr', 'MS', 'Auburn University', 7, 110);

INSERT INTO VETERINARIAN VALUES(54234, 'Aaron', 'Schmidt', 'CT', 'University of Pennsylvania', 4, 101);

INSERT INTO VETERINARIAN VALUES(10924, 'Abel', 'Malone', 'TX', 'Tulane University', 10, 103);

INSERT INTO VETERINARIAN VALUES(78039, 'Hannah', 'Farrington', 'KS', 'University of Oklahoma', 2, 100);

INSERT INTO VETERINARIAN VALUES(20341, 'Jason', 'Connolly', 'MD', 'Johns Hopkins University', 5, 104);

INSERT INTO VETERINARIAN VALUES(50987, 'Steven', 'Koch', 'TN', 'Vanderbilt University', 12,102);

INSERT INTO VETERINARIAN VALUES(69028, 'Kyle', 'Hautin', 'TX', 'Texas Tech University', 10, 106);

INSERT INTO VETERINARIAN VALUES(50239, 'Joyce', 'Howe', 'NC', 'Wake Forest University', 4, 107);

INSERT INTO VETERINARIAN VALUES(40193, 'Anthony', 'Meyers', 'PA', 'University of Pittsburgh', 35, 101);

INSERT INTO VETERINARIAN VALUES(11596, 'Ethan', 'Fuller', 'OH', 'Columbia University', 5, 114);

INSERT INTO VETERINARIAN VALUES(74432, 'Samantha', 'Gutierrez', 'NH', 'Harvard University', 12, 101);

INSERT INTO VETERINARIAN VALUES(32009, 'Rebecca', 'Henderson', 'FL', 'University of Miami', 11, 113);

INSERT INTO VETERINARIAN VALUES(43011, 'Louis', 'Greene', 'NY', 'Columbia University', 29, 104);

INSERT INTO VETERINARIAN VALUES(10023, 'Joseph', 'York', 'IN', 'Purdue University', 11, 112);

INSERT INTO VETERINARIAN VALUES(45587, 'Maria', 'Ebbs', 'NH', 'Columbia University', 21, 102);

INSERT INTO VETERINARIAN VALUES(56678, 'Theo', 'Hughes', 'AL', 'Auburn University', 25, 108);

INSERT INTO VETERINARIAN VALUES(22399, 'Ashley', 'Morrison', 'OK', 'Oklahoma State University', 30, 109);

INSERT INTO VETERINARIAN VALUES(57899, 'Elliot', 'Shaw', 'NJ', 'Cornell University', 19, 111);

/\*Add Records to Species Table \*/

INSERT INTO SPECIES VALUES(300, 'DOG', 10.0, 13.0);

INSERT INTO SPECIES VALUES(301, 'CAT', 13.0, 17.0);

INSERT INTO SPECIES VALUES(302, 'HAMSTER', 1.0, 3.5);

INSERT INTO SPECIES VALUES(303, 'SNAKE', 15.0, 30.0);

INSERT INTO SPECIES VALUES(304, 'BIRD', 10.0, 50.0);

INSERT INTO SPECIES VALUES(305, 'HORSE', 25.0, 30.0);

INSERT INTO SPECIES VALUES(306, 'PIG', 12.0, 18.0);

INSERT INTO SPECIES VALUES(307, 'CHICKEN', 8.0, 15.0);

INSERT INTO SPECIES VALUES(308, 'COW', 18.0, 22.0);

INSERT INTO SPECIES VALUES(309, 'GUINEA PIG', 4.0, 8.0);

INSERT INTO SPECIES VALUES(310, 'GOAT', 8.0, 12.0);

INSERT INTO SPECIES VALUES(311, 'RABBIT', 5.0, 10.0);

INSERT INTO SPECIES VALUES(312, 'FISH', 3.0, 5.0);

INSERT INTO SPECIES VALUES(313, 'RAT', 2.0, 4.0);

INSERT INTO SPECIES VALUES(314, 'FERRET', 5.0, 9.0);

/\*Add Records to Treatment Table \*/

INSERT INTO TREATMENT VALUES(200, 'Chemo');

INSERT INTO TREATMENT VALUES(201, 'Heart Surgery');

INSERT INTO TREATMENT VALUES(202, 'Antibiotics');

INSERT INTO TREATMENT VALUES(203, 'Anti-inflammatory Drugs');

INSERT INTO TREATMENT VALUES(204, 'Eye Drops');

INSERT INTO TREATMENT VALUES(205, 'Diet Change')

INSERT INTO TREATMENT VALUES(206, 'Bone Replacement');

INSERT INTO TREATMENT VALUES(207, 'Topical Treatment');

INSERT INTO TREATMENT VALUES(208, 'Antifungal Agent');

INSERT INTO TREATMENT VALUES(209, 'Ureteroscopy');

INSERT INTO TREATMENT VALUES(210, 'Mouth Rinse');

INSERT INTO TREATMENT VALUES(211, 'Gauze');

INSERT INTO TREATMENT VALUES(212, 'Oxygen Therapy');

INSERT INTO TREATMENT VALUES(213, 'Fluid Therapy');

INSERT INTO TREATMENT VALUES(214, 'Deworming Drugs');

/\*Add Records to Insurance Table \*/

INSERT INTO INSURANCE VALUES(500, 'Jean', 'Harvey', 'Spot Pet Insurance', 250);

INSERT INTO INSURANCE VALUES(501, 'Charlotte', 'Ray', 'Healthy Paws', 750);

INSERT INTO INSURANCE VALUES(502, 'Rachel', 'Neal', 'Fetch', 1000);

INSERT INTO INSURANCE VALUES(503, 'Saul', 'Olson', 'Spot Pet Insurance', 75);

INSERT INTO INSURANCE VALUES(504, 'Jasmine', 'Powell', 'Nationwide', 950);

INSERT INTO INSURANCE VALUES(505, 'Chris', 'Ward', 'Lemonade', 150);

INSERT INTO INSURANCE VALUES(506, 'Brandy', 'Wagner', 'Pumpkin', 450);

INSERT INTO INSURANCE VALUES(507, 'Violet', 'Lee', 'Healthy Paws', 500);

INSERT INTO INSURANCE VALUES(508, 'Iris', 'Lambert', 'Fetch', 1000);

INSERT INTO INSURANCE VALUES(509, 'Melissa', 'Hoch', 'Nationwide', 200);

INSERT INTO INSURANCE VALUES(510, 'Marcus', 'Erickson', 'Lemonade', 175);

INSERT INTO INSURANCE VALUES(511, 'Seth', 'Fletcher', 'MetLife', 50);

INSERT INTO INSURANCE VALUES(512, 'Rickey', 'Blake', 'Fetch', 225);

INSERT INTO INSURANCE VALUES(513, 'Patty', 'Phelps', 'Spot Pet Insurance', 450);

INSERT INTO INSURANCE VALUES(514, 'Nora', 'Shaw', 'Healthy Paws', 750);

/\*Add Records to PETOWNER Table \*/

INSERT INTO PETOWNER values(600, 'Harry', 'Clawson', '614-540-9068');

INSERT INTO PETOWNER values(601, 'Melissa', 'Hoch', '542-091-5403');

INSERT INTO PETOWNER values(602, 'Jean', 'Harvey', '614-543-2014');

INSERT INTO PETOWNER values(603, 'Iris', 'Lambert', '430-251-5403');

INSERT INTO PETOWNER values(604, 'Violet', 'Lee', '430-605-4034');

INSERT INTO PETOWNER values(605, 'Derek', 'Padilla', '645-039-5430');

INSERT INTO PETOWNER values(606, 'Patty', 'Phelps', '542-016-3402');

INSERT INTO PETOWNER values(607, 'Saul', 'Olson', '614-960-4345');

INSERT INTO PETOWNER values(608, 'Charlotte', 'Ray', '430-321-6505');

INSERT INTO PETOWNER values(609, 'Nora', 'Shaw', '542-998-5569');

INSERT INTO PETOWNER values(610, 'Rickey', 'Blake', '614-332-5044');

INSERT INTO PETOWNER values(611, 'Rachel', 'Neal', '542-569-3029');

INSERT INTO PETOWNER values(612, 'Jean', 'Harvey', '645-789-0342');

INSERT INTO PETOWNER values(613, 'Marcus', 'Erickson', '645-777-6507');

INSERT INTO PETOWNER values(614, 'Jamie', 'Flores', '812-043-2910');

INSERT INTO PETOWNER values(615, 'Brandy', 'Wagner', '430-860-3920');

INSERT INTO PETOWNER values(616, 'Seth', 'Fletcher', '615-430-5552');

INSERT INTO PETOWNER values(617, 'Chris', 'Ward', '542-224-5320');

INSERT INTO PETOWNER values(618, 'Jasmine', 'Powell', '653-342-4441');

/\*Add Records to Patient Table \*/

INSERT INTO PATIENT VALUES(400, 'Ranger', 'Lyme Disease', '2012-03-02', 300, 45587, 513, 618);

INSERT INTO PATIENT VALUES(401, 'Whinny', 'Tetanus', '1999-12-26', 305, 50987, NULL, 600);

INSERT INTO PATIENT VALUES(402, 'Pinky', 'Conjunctivitis', '2019-07-06', 306, 43011, NULL, 612);

INSERT INTO PATIENT VALUES(403, 'Tabby', 'Gastritis', '2004-05-21', 301, 69504, 508, 602);

INSERT INTO PATIENT VALUES(404, 'Chipper', 'Lymphoma', '2012-08-31', 304, 78039, 510, 610);

INSERT INTO PATIENT VALUES(405, 'Dixie', 'Cardiomyopathy', '2017-04-14', 307, 74432, NULL, 604);

INSERT INTO PATIENT VALUES(406, 'Gizmo', 'Urticaria', '2022-10-01', 302, 43890, 511, 608);

INSERT INTO PATIENT VALUES(407, 'Ebony', 'Ear Mites', '2011-05-16', 301, 11596, 505, 606);

INSERT INTO PATIENT VALUES(408, 'Bella', 'Arrhythmia', '2015-02-17', 300, 54234, 504, 616);

INSERT INTO PATIENT VALUES(409, 'Yuki', 'Asthma', '2006-08-18', 301, 10924, 514, 601);

INSERT INTO PATIENT VALUES(410, 'Nacho', 'Uveitis', '1989-06-08', 300, 20341,500, 600);

INSERT INTO PATIENT VALUES(411, 'Rex', 'Hip Dysplasia', '2014-01-10', 311, 50239, 507, 617);

INSERT INTO PATIENT VALUES(412, 'Benji', 'Periodontitis', '2010-11-18', 308, 69028, 501, 614);

INSERT INTO PATIENT VALUES(413, 'Jasper', 'Atherosclerosis', '1996-09-17', 304, 40193, 512, 610);

INSERT INTO PATIENT VALUES(414, 'Earl', 'Ringworm', '2009-02-23', 310, 32009, 506, 603);

INSERT INTO PATIENT VALUES(415, 'Hamlet', 'Urolith', '2014-09-13', 309, 10023, NULL, 605);

INSERT INTO PATIENT VALUES(416, 'Skye', 'Fin Rot', '2020-11-05', 312, 45587, 502, 609);

INSERT INTO PATIENT VALUES(417, 'Brownie', 'Bumblefoot', '2018-04-12', 313, 56678, NULL, 611);

INSERT INTO PATIENT VALUES(418, 'Felix', 'Stroke', '2007-12-14', 301, 22399, 503, 613);

INSERT INTO PATIENT VALUES(419, 'Hopper', 'Kidney Cyst', '2016-07-22', 311, 57899, 509, 615);

INSERT INTO PATIENT VALUES(420, 'Zippy', 'Roundworm', '2022-05-19', 314, 32009, NULL, 607);

/\*Add Records to APPOINTMENT Table \*/

INSERT INTO APPOINTMENT values(700, 400, '20230202', '2:30pm', '3:15pm', 202);

INSERT INTO APPOINTMENT values(701, 401, '20221204', '11:10am', '1:00pm', 202);

INSERT INTO APPOINTMENT values(702, 402, '20221105', '3:00pm', '4:45pm', 204);

INSERT INTO APPOINTMENT values(703, 403, '20230101', '11:00am', '11:45am', 205);

INSERT INTO APPOINTMENT values(704, 404, '20230405', '7:30 am', '9:25am', 200);

INSERT INTO APPOINTMENT values(705, 405, '20230325','8:25am', '10:15am', 201);

INSERT INTO APPOINTMENT values(706, 406, '20221030', '11:45am', '12:45pm', 203);

INSERT INTO APPOINTMENT values(707, 407, '20230117', '7:00am', '8:32am', 207);

INSERT INTO APPOINTMENT values(708, 408, '20221228', '10:15am', '12:00pm', 201);

INSERT INTO APPOINTMENT values(709, 409, '20230304', '9:15am', '9:45am', 203);

INSERT INTO APPOINTMENT values(710, 410, '20221114', '7:55am', '9:26am', 204);

INSERT INTO APPOINTMENT values(711, 411, '20230225', '8:24am', '9:35am', 206);

INSERT INTO APPOINTMENT values(712, 412, '20230119','9:45am', '10:06am', 201);

INSERT INTO APPOINTMENT values(713, 413, '20230327', '8:02am', '9:16am', 201);

INSERT INTO APPOINTMENT values(714, 414, '20221114', '7:13am', '8:43am', 208);

INSERT INTO APPOINTMENT values(715, 415, '20221201', '6:45am', '7:40am', 209);

INSERT INTO APPOINTMENT values(716, 416, '20230322', '8:06am', '10:15am', 208);

INSERT INTO APPOINTMENT values(717, 417, '20221103', '9:15am', '10:55am', 211);

INSERT INTO APPOINTMENT values(718, 418, '20230215', '2:45pm', '3:55pm', 212);

INSERT INTO APPOINTMENT values(719, 419, '20230402', '3:15pm', '5:00pm', 213);

INSERT INTO APPOINTMENT values(720, 420, '20230118', '1:30pm', '3:00pm', 214);

INSERT INTO APPOINTMENT values(721, 409, '20161205', '11:30am', '3:00pm', 201);

INSERT INTO APPOINTMENT values(722, 403, '20170321', '9:00am', '10:00am', 206);

**II. Queries**

1. **All CREATE TABLES are provided above with PK and FK constraints, where necessary.**
2. **An UPDATE query**

**UPDATE PETOWNER**

**SET PHONE\_NUMBER = '430-699-6798'**

**WHERE PET\_OWNER\_ID = 617**

This query changes the phone number of Chris Ward, whose Pet Owner ID is 617 to ‘430-699-6798.’ This could be a useful query for whenever clients of the Vet Clinic get new phone numbers.

**2. (cont). An ALTER query**

**ALTER TABLE INSURANCE**

**ALTER COLUMN POLICY\_HOLDER\_FNAME varchar(35)**

This query changes the data type of the first name of the pet insurance policy holder. Previously, this column could hold first names with up to 20 characters, leaving blank space holders for any first names that were less than 20 characters. After this alter query, the first name field in the insurance table could hold up to 35 characters, which is more accommodating since it could be possible that a first name is longer than 20 characters.

1. **A simple SELECT query with a WHERE clause**

**SELECT \* FROM**

**VETERINARIAN**

**WHERE STATE\_LICENSE <> 'TX'**

This query selects all the fields in the veterinarian table (VET\_ID, VET\_FNAME, VET\_LNAME, STATE\_LICENSE, PHD\_SCHOOL, YRS\_PRACTICE, and EXPERTISE\_ID) where the veterinarian is licensed to practice in any state except for Texas.

1. **A query with a mask (the LIKE clause or similar)**

**SELECT OWNER\_FNAME, OWNER\_LNAME, PHONE\_NUMBER**

**FROM PETOWNER**

**WHERE PHONE\_NUMBER LIKE '430%' OR PHONE\_NUMBER LIKE '614%'**

This query selects the pet owner’s first name, last name, and phone number for all pet owners whose phone numbers start with the numbers ‘430’ or ‘614.’ This query could be useful in retrieving the names and phone numbers of pet owners with certain area codes.

1. **At least one query contains an ORDER BY clause**

**SELECT PATIENT\_NAME, OWNER\_FNAME, OWNER\_LNAME, COMMON\_NAME, LIFESPAN\_UPPER\_YEARS**

**FROM SPECIES S JOIN PATIENT P**

**ON S.SPECIES\_ID = P.SPECIES\_ID**

**JOIN PETOWNER O**

**ON O.PET\_OWNER\_ID = P.PET\_OWNER\_ID**

**WHERE LIFESPAN\_UPPER\_YEARS > 10**

**ORDER BY LIFESPAN\_UPPER\_YEARS**

This query returns the name of the animal, the owner’s first and last name, the type of animal, and the value at the higher range of the expected lifespan for that specific animal for those animals that have an expected lifespan of greater than 10 years at the high end of the expected lifespan range. The output is ordered from low to high by the upper lifespan values in years.

1. **A query with an aggregate function**

**SELECT COUNT(BIRTHDATE) AS 'Number of Animals with birthdays between 2016 and 2020'**

**FROM PATIENT**

**WHERE BIRTHDATE > '2016-01-01' AND BIRTHDATE < '2020-12-31'**

This query returns the number of animals in the “Priority Paws’ Veterinary Database who were born after January 1, 2016 but before December 31, 2020. This would be a useful query structure to adjust to determine the number of animals born before, after, or between any dates.

1. **A query with a GROUP BY clause**

**SELECT MIN((CONVERT(TIME, (CAST(END\_TIME AS datetime) - CAST(BEG\_TIME AS datetime)))))**

**AS ELAPSED\_TIME,TREATMENT\_NAME**

**FROM APPOINTMENT**

**JOIN TREATMENT ON APPOINTMENT.TREATMENT\_ID = TREATMENT.TREATMENT\_ID**

**GROUP BY(TREATMENT\_NAME)**

This query returns the minimum elapsed appointment time for each treatment. The query allows times to be treated as a date time format with SQL-defined standard date of January 1, 1900 for the datetime function. This date is irrelevant because we are only worried about the magnitude of time between two different times on the same day. The datetime format just allows us to subtract the two times unlike the “time” data type. After the elapsed times are determined, the query also selects the names of the treatments by joining the appointment table and treatment table on the common field of TREATMENT\_ID so that the output is the minimum appointment time in the database for each of the treatments for the data recorded of animal appointments. For example, you could find the quickest appointment from all records of animals who were seen and ended up receiving antibiotics as a treatment.

1. **A query with a calculated field**

**SELECT VET\_FNAME, VET\_LNAME,**

**PHD\_SCHOOL, 47 - YRS\_PRACTICE AS 'YEARS\_UNTIL\_RETIREMENT'**

**FROM VETERINARIAN**

**WHERE(47-YRS\_PRACTICE) >= 0**

This query selects the first and last name of a veterinarian, the school from which they received their PHD, and the number of estimated years they have until retirement, using a general assumption that a vet will work 47 years until they retire. This query will only return the years until retirement using this assumption if the vet has 47 or less years of experience. In the records I inserted into the VETERINARIAN table, all vets have less than 47 years of experience, so this query returns fields for all vets, but it is good practice to use the WHERE clause in case there was a vet who has worked for over 47 years and not retired yet.

1. **A query with GROUP BY / HAVING**

**SELECT COMMON\_NAME, COUNT(PET\_OWNER\_ID) AS 'Number of Owners with this Animal'**

**FROM PATIENT P JOIN SPECIES S**

**ON P.SPECIES\_ID = S.SPECIES\_ID**

**GROUP BY COMMON\_NAME**

**HAVING (COUNT(PET\_OWNER\_ID) >1)**

This query returns the common name of animals in the vet database along with the number of owners who have each type of animal currently in the database. These results are returned only for animal types where there is more than one patient of that type. The query counts the number of times a pet owner ID occurs in the patient table and traces this to the common name by joining the patient and species tables on the common field of SPECIES\_ID. Since birds, cats, dogs, and rabbits are the only animals entered as patients more than one time (2 times, 4 times, 3 times, and 2 times, respectively), these animals are the only ones reported under the COMMON\_NAME field, along with the values above in the column aliased “Number of Owners with this Animal.”

1. **A query with a subquery**

**SELECT P.PATIENT\_ID, PATIENT\_NAME, DIAGNOSIS**

**FROM PATIENT P**

**JOIN APPOINTMENT A ON P.PATIENT\_ID = A.PATIENT\_ID**

**JOIN TREATMENT T ON T.TREATMENT\_ID = A.TREATMENT\_ID**

**WHERE T.TREATMENT\_ID IN (SELECT TREATMENT\_ID FROM TREATMENT**

**WHERE TREATMENT\_NAME LIKE '%Therapy'**

**OR TREATMENT\_NAME LIKE '%Therapy')**

This subquery returns the ID number, name, and diagnosis of all patients who received a treatment that literally ends in the word “Therapy.” This is because the treatments of oxygen therapy and fluid therapy are currently inserted into the treatment table. So, to see all patients who received either therapy, or any future treatments that end in therapy, we must use the IN operator to compare the treatment ID values to a list of treatment IDs, rather than one. Since we do not know the treatment ID values beforehand that will return a treatment name ending in therapy, we must use an IN subquery to select the treatment IDs that will be in the list for comparison to TREATMENT\_ID. This query uses appointment table as a bridge between patient and treatment by means of the PATIENT\_ID field.

III. Views

**View # 1**

**CREATE VIEW recent\_appointments AS**

**SELECT APT\_ID, PATIENT\_NAME, APT\_DATE, BEG\_TIME, END\_TIME**

**FROM APPOINTMENT A JOIN PATIENT P**

**ON A.PATIENT\_ID = P.PATIENT\_ID**

**WHERE APT\_DATE >= '2022-12-01'**

This view shows up-to-date results of recent appointments that were held after December 1, 2022. It shows the appointment ID, the patient name, the date of the appointment, and the beginning and ending times for those appointments that occurred only after December 1, 2022 with base tables of patient and appointment.

**View # 2**

**CREATE VIEW auburn\_alumni AS**

**SELECT VET\_FNAME, VET\_LNAME, STATE\_LICENSE, AREA\_OF\_EXPERTISE**

**FROM VETERINARIAN V JOIN PATIENT P ON V.VET\_ID = P.VET\_ID**

**JOIN EXPERTISE E ON E.EXPERTISE\_ID = V.EXPERTISE\_ID**

**WHERE PHD\_SCHOOL = 'Auburn University'**

This view returns the full name of veterinarians, the state they are licensed in, and their area of expertise for only the veterinarians who earned their PHD from Auburn University.

**IV. Joins**

1. **Simple Join with a WHERE clause**

**PURPOSE: This query selects the animal’s patient ID number, the animal’s name, and the deductible of its insurance policy in dollars for all animals with pet insurance deductibles less than $500.**

**SELECT PATIENT\_ID, PATIENT\_NAME, DEDUCTIBLE\_DOLLARS**

**FROM PATIENT**

**JOIN INSURANCE**

**ON PATIENT.POLICY\_NO = INSURANCE.POLICY\_NO**

**WHERE DEDUCTIBLE\_DOLLARS < 500**

1. **Any join of your choice that includes 3 or more tables.**

**PURPOSE: This query selects the patient ID number, patient name, the first and last name of the vet taking care of each patient, and the diagnosis for only the patients that are dogs in the database.**

**SELECT PATIENT\_ID, PATIENT\_NAME, VET\_FNAME, VET\_LNAME, DIAGNOSIS**

**FROM PATIENT JOIN**

**SPECIES**

**ON PATIENT.SPECIES\_ID = SPECIES.SPECIES\_ID**

**JOIN**

**VETERINARIAN**

**ON**

**PATIENT.VET\_ID = VETERINARIAN.VET\_ID**

**WHERE COMMON\_NAME = 'DOG'**

1. **One of the Following: UNION, UNION ALL, INTERSECT, MINUS**

**PURPOSE: This query selects the first and last names of only the pet owners that have insurance policies for their pet.**

**SELECT POLICY\_HOLDER\_FNAME, POLICY\_HOLDER\_LNAME**

**FROM INSURANCE**

**INTERSECT**

**SELECT OWNER\_FNAME, OWNER\_LNAME**

**FROM PETOWNER**

1. **INNER JOIN**

**Purpose: This query provides the name of an animal with their diagnosis, the date of their appointment, and the beginning and ending times of their appointment for only the animals whose appointments ended later than 11:00 am.**

**SELECT PATIENT\_NAME, DIAGNOSIS,**

**APT\_DATE, BEG\_TIME, END\_TIME**

**FROM PATIENT**

**INNER JOIN**

**APPOINTMENT**

**ON PATIENT.PATIENT\_ID = APPOINTMENT.PATIENT\_ID**

**WHERE(END\_TIME > (SELECT CONVERT(TIME, '11:00am')))**

1. **LEFT OUTER JOIN**

**Purpose: This query returns the patient ID, patient name, birthday, policy number and insurance company name for all of the animal patients, even those who are not covered with insurance and have no policy number or insurance company in the right table.**

**SELECT PATIENT\_ID, PATIENT\_NAME, BIRTHDATE, PATIENT.POLICY\_NO,**

**INSURANCE\_CO**

**FROM PATIENT LEFT OUTER JOIN INSURANCE ON**

**PATIENT.POLICY\_NO = INSURANCE.POLICY\_NO**

**V. Stored Procedures**

**Stored Procedure #1 called InsertPetOwner**

**CREATE PROCEDURE InsertPetOwner(**

**@PET\_OWNER\_ID int,**

**@OWNER\_FNAME varchar(20),**

**@OWNER\_LNAME varchar(35),**

**@PHONE\_NUMBER char(12))**

**AS**

**INSERT INTO PETOWNER VALUES(@PET\_OWNER\_ID,**

**@OWNER\_FNAME, @OWNER\_LNAME, @PHONE\_NUMBER)**

/\* EXAMPLE OF INSERTING LINE USING STORED PROCEDURE \*/

EXEC InsertPetOwner 619, 'Ellie', 'Blane', '654-908-3221'

This stored procedure adds a new pet owner to the PETOWNER table, passing all the field values as procedures. The bottom line executes the stored procedure, and I made sure this line was added to the PETOWNER table. EXEC can add new pet owners to the PETOWNER table without having to run additional code, such as an INSERT INTO query.

**STORED PROCEDURE #2 called UpdateVetInfo**

**CREATE PROCEDURE UpdateVetInfo(**

**@VET\_ID int,**

**@VET\_FNAME varchar(20), @VET\_LNAME varchar(30), @STATE\_LICENSE char(2),**

**@PHD\_SCHOOL varchar(100), @YRS\_PRACTICE int, @EXPERTISE\_ID int)**

**AS**

**UPDATE VETERINARIAN SET VET\_FNAME = @VET\_FNAME, VET\_LNAME = @VET\_LNAME,**

**STATE\_LICENSE = @STATE\_LICENSE, PHD\_SCHOOL = @PHD\_SCHOOL, YRS\_PRACTICE = @YRS\_PRACTICE,**

**EXPERTISE\_ID = @EXPERTISE\_ID**

**WHERE VET\_ID = @VET\_ID**

/\*EXAMPLE OF UPDATING LINE USING STORED PROCEDURE \*/

EXEC UpdateVetInfo 43011, 'Gracie', 'Lopez', 'TX', 'Alabama State', 12, 100

This stored procedure adjusts the full name of a vet, his or her state license, the school where they received a PHD, the years of experience they have, and their area of expertise for a specific VET ID code. All variables are passed as parameters, and every value other than VET\_ID changes, as VET\_ID is used in the WHERE statement to dictate under which vet ID code all the other variables should be changed according to their parameter values. The EXEC statement given as an example above would change the fields related to VET\_ID = 43011, which was previously assigned to be Louis Greene in the original INSERT INTO tables. This would be useful if the vet clinic recycles vet ID numbers after a veterinarian quits, as the employee name and corresponding variables could change while keeping the VET ID number the same.

**STORED PROCEDURE #3 Called DeleteAppointment**

**CREATE PROCEDURE DeleteAppointment(**

**@PATIENT\_ID int, @APT\_DATE DATE, @BEG\_TIME varchar(8), @END\_TIME varchar(8))**

**AS**

**DELETE FROM APPOINTMENT**

**WHERE PATIENT\_ID = @PATIENT\_ID AND APT\_DATE = @APT\_DATE AND BEG\_TIME = @BEG\_TIME AND END\_TIME = @END\_TIME**

This stored procedure deletes a record from the Appointment table according for a specified combination of Patient ID number, date of appointment, a beginning appointment time, and an ending appointment time, which are the parameters. This would allow for not all appointments for a patient ID number to be deleted, but just those for certain dates and times.

For example, running the line below would delete the record for a patient whose ID number is 400 and who had an appointment on February 2, 2023 from 2:30 pm to 3:15 pm:

EXEC DeleteAppointment 400, '20230202', '2:30pm', '3:15pm'

VI. A trigger

**\\* Creates a backup table for APPOINTMENT \*\**

CREATE TABLE APPOINTMENT\_BACKUP(

APT\_ID int, PATIENT\_ID int, APT\_DATE date,

BEG\_TIME varchar(8), END\_TIME varchar(8),

TREATMENT\_ID int

);

**/\* Creates a trigger for whenever a record is deleted from the appointment table \*/**

CREATE TRIGGER AfterDeleteAppointment on APPOINTMENT

FOR DELETE

AS DECLARE @apt\_id int,

@patient\_id int,

@apt\_date date,

@beg\_time varchar(8),

@end\_time varchar(8),

@treatment\_id int;

SELECT @apt\_id = del.APT\_ID FROM DELETED del;

SELECT @patient\_id = del.PATIENT\_ID FROM DELETED del;

SELECT @apt\_date = del.APT\_DATE FROM DELETED del;

SELECT @beg\_time = del.BEG\_TIME FROM DELETED del;

SELECT @end\_time = del.END\_TIME FROM DELETED del;

SELECT @treatment\_id = del.TREATMENT\_ID FROM DELETED del;

**/\* Ensures deleted record from APPOINTMENT gets added to APPOINTMENT\_BACKUP \*/**

INSERT INTO APPOINTMENT\_BACKUP

VALUES(@apt\_id, @patient\_id, @apt\_date,

@beg\_time, @end\_time, @treatment\_id)

**/\*Tests to make sure that this record will be deleted from APPOINTMENT and added to APPOINTMENT\_BACKUP \*/**

DELETE FROM APPOINTMENT

WHERE APT\_ID = 722

This trigger automatically adds a record to the table called APPOINTMENT\_BACKUP whenever it is deleted from the APPOINTMENT table. The bottom query deleted the record with an appointment ID of 722 from the appointment table, and it can be confirmed that this record is now in the APPOINTMENT\_BACKUP table