

Fido’s Store Database

Brandon Dave, Samantha Stetz

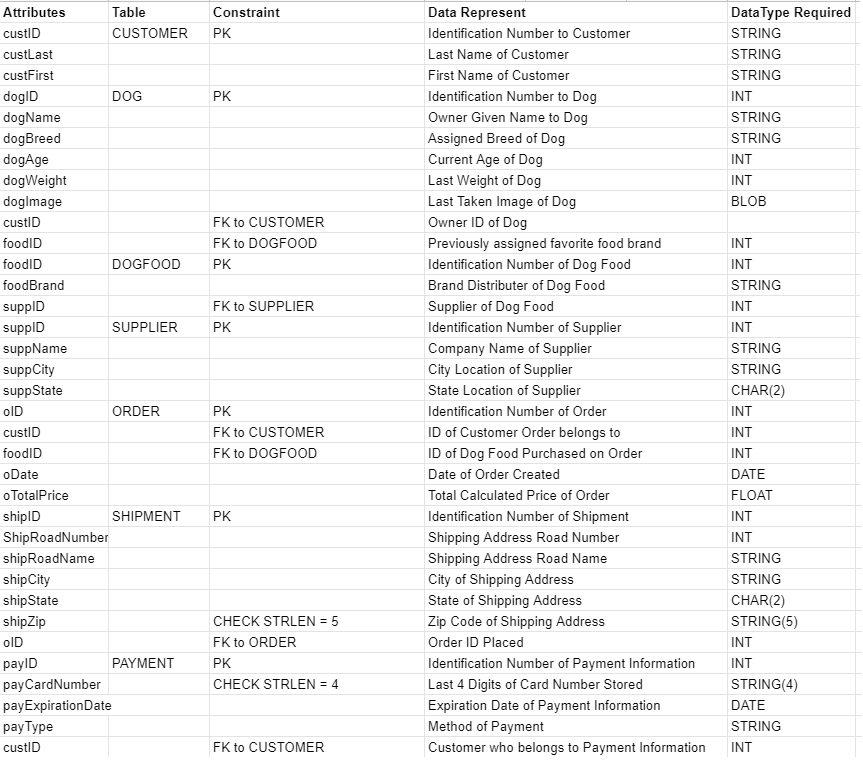
CS3700: FP10

Project Description

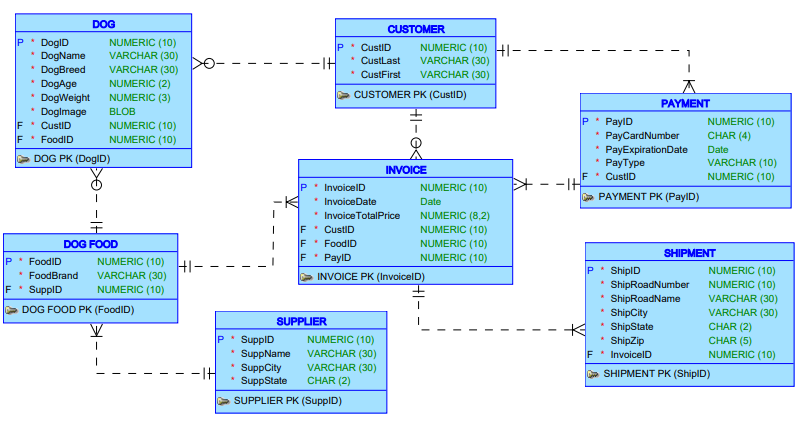


The purpose of the Fido Store Database is to store information about customers and their online dog food orders. To begin, the database contains information about the different types of dog food, which includes information about the brand and supplier. These dog foods will be the favorite food of many different dogs. However, a dog can only have one favorite type of dog food, if they have a favorite food at all. The dogs have basic information listed such as their age, weight, and breed. There is also the opportunity to upload a picture of the dog. Although a dog can only have one owner, an owner may have many dogs. Even though an owner can list many dogs, they are not required to list any. Owners, which are known as customers in the database, only have very basic information stored, which includes their name and identification number. Customers can place orders, which creates an invoice detailing information such as the date and total price of the dog food order. An invoice will only have one customer listed on it. The many types of dog food that a customer can order are supplied by many different suppliers, which have their name and location listed. A type of dog food will come from one specific supplier. The order placed will require a single method of payment, which means that information such as the last four numbers on the card, the expiration date, and the pay type must be given by the customer. A customer can use many different types of payments over many different orders. However, a payment can only have one customer listed. Finally, once the order has been processed, it will eventually be shipped. Orders can be shipped many different places, but a shipment will only have one order attached to it. The shipment notice will contain information about the customer’s specified address.

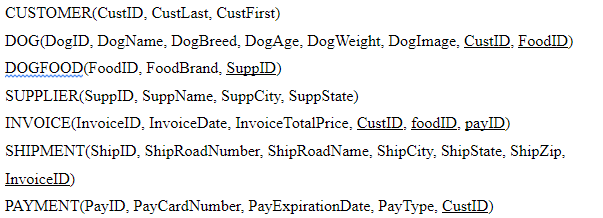
Data Dictionary



Entity-Relationship Diagram



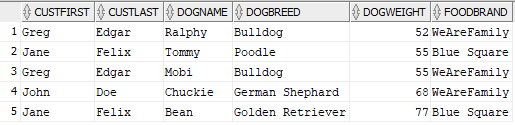
Schema



SQL Script Output

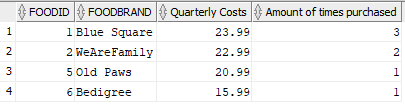
Three Table Join

* Display the owners of dogs and the respective dog’s favorite food that are over 50 pounds



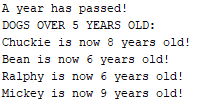
Two Table Statistical Join with a Single Row Function

* Display unique dog food brands purchased in the last 3 months along with their average cost and total point of times they were purchased



Automated PL/SQL

* Automate Dog Age increment and Display Dog’s over the age of 5



SQL Script Commands

-- Brandon Dave, Samantha Stetz, Muhammad Syed

-- CS3700: Final Project

-- The entered data is fictional.

-- Drop Commands

DROP TABLE SHIPMENT;

DROP TABLE INVOICE;

DROP TABLE PAYMENT;

DROP TABLE DOG;

DROP TABLE DOGFOOD;

DROP TABLE SUPPLIER;

DROP TABLE CUSTOMER;

DROP MATERIALIZED VIEW VWCUSTOMERDOGFAVORITEFOOD;

DROP MATERIALIZED VIEW VWCUSTOMERINVOICEDOGFOOD;

-- Create Commands

CREATE TABLE CUSTOMER(

CustID NUMERIC(10) CONSTRAINT customer\_CustID\_pk PRIMARY KEY,

CustLast VARCHAR(30),

CustFirst VARCHAR(30)

);

CREATE TABLE SUPPLIER(

SuppID NUMERIC(10) CONSTRAINT supplier\_SuppID\_pk PRIMARY KEY,

SuppName VARCHAR(30),

SuppCity VARCHAR(30),

SuppState CHAR(2)

);

CREATE TABLE DOGFOOD(

FoodID NUMERIC(10) CONSTRAINT dogfood\_FoodID\_pk PRIMARY KEY,

FoodBrand VARCHAR(30),

SuppID NUMERIC(10) CONSTRAINT dogfood\_SuppID\_fk REFERENCES SUPPLIER(SuppID)

);

CREATE TABLE DOG (

DogID NUMERIC(10) CONSTRAINT dog\_DogID\_pk PRIMARY KEY,

DogName VARCHAR(30),

DogBreed VARCHAR(30),

DogAge NUMERIC(2),

DogWeight NUMERIC(3),

DogImage BLOB,

CustID NUMERIC(15) CONSTRAINT dog\_CustID\_fk REFERENCES CUSTOMER(CustID),

FoodID NUMERIC(10) CONSTRAINT dog\_FoodID\_fk REFERENCES DOGFOOD(FoodID)

);

CREATE TABLE PAYMENT (

PayID NUMERIC(10) CONSTRAINT payment\_PayID\_pk PRIMARY KEY,

PayCardNumber CHAR(4),

PayExpirationDate DATE,

PayType VARCHAR(30),

CustID NUMERIC(10) CONSTRAINT payment\_CustID\_fk REFERENCES CUSTOMER(CustID)

);

CREATE TABLE INVOICE(

InvoiceID NUMERIC(10) CONSTRAINT invoice\_InvoiceID\_pk PRIMARY KEY,

InvoiceDate DATE,

InvoiceTotalPrice NUMERIC(6,2),

CustID NUMERIC(15) CONSTRAINT invoice\_CustID\_fk REFERENCES CUSTOMER(CustID),

FoodID NUMERIC(10) CONSTRAINT invoice\_FoodID\_fk REFERENCES DOGFOOD(FoodID),

PayID NUMERIC(10) CONSTRAINT invoice\_PayID\_fk REFERENCES PAYMENT(PayID)

);

CREATE TABLE SHIPMENT (

ShipID NUMERIC(10) CONSTRAINT shipment\_ShipID\_pk PRIMARY KEY,

ShipRoadNumber NUMERIC(10),

ShipRoadName VARCHAR(30),

ShipCity VARCHAR(30),

ShipState CHAR(2),

ShipZip CHAR(5),

InvoiceID NUMERIC(10) CONSTRAINT shipment\_InvoiceID\_fk REFERENCES INVOICE(InvoiceID)

);

-- Insert Commands

-- CUSTOMER(CustID, CustLast, CustFirst)

INSERT INTO CUSTOMER(CustID, CustLast, CustFirst)

VALUES ('01', 'Doe', 'John');

INSERT INTO CUSTOMER(CustID, CustLast, CustFirst)

VALUES ('02', 'Felix', 'Jane');

INSERT INTO CUSTOMER(CustID, CustLast, CustFirst)

VALUES ('03', 'Edgar', 'Greg');

INSERT INTO CUSTOMER(CustID, CustLast, CustFirst)

VALUES ('04', 'Euler', 'Matt');

INSERT INTO CUSTOMER(CustID, CustLast, CustFirst)

VALUES ('05', 'Proof', 'Kristin');

-- SUPPLIER(SuppID, SuppName, SuppCity, SuppState)

INSERT INTO SUPPLIER(SuppID, SuppName, SuppCity, SuppState)

VALUES ('01', 'Barkers Supplies', 'Fairborn', 'OH');

INSERT INTO SUPPLIER(SuppID, SuppName, SuppCity, SuppState)

VALUES ('02', 'Dog Co', 'Beavercreek', 'OH');

INSERT INTO SUPPLIER(SuppID, SuppName, SuppCity, SuppState)

VALUES ('03', 'Paws and More', 'Dayton', 'OH');

-- DOGFOOD(FoodID, FoodBrand, SuppID)

INSERT INTO DOGFOOD(FoodID, FoodBrand, SuppID)

VALUES('01', 'Blue Square', '01');

INSERT INTO DOGFOOD(FoodID, FoodBrand, SuppID)

VALUES('02', 'WeAreFamily', '01');

INSERT INTO DOGFOOD(FoodID, FoodBrand, SuppID)

VALUES('03', 'Furrina', '02');

INSERT INTO DOGFOOD(FoodID, FoodBrand, SuppID)

VALUES('04', 'Special Paws', '02');

INSERT INTO DOGFOOD(FoodID, FoodBrand, SuppID)

VALUES('05', 'Old Paws', '02');

INSERT INTO DOGFOOD(FoodID, FoodBrand, SuppID)

VALUES('06', 'Bedigree', '03');

-- DOG(DogID, DogName, DogBreed, DogAge, DogWeight, DogImage, CustID, FoodID)

INSERT INTO DOG(DogID, DogName, DogBreed, DogAge, DogWeight, DogImage, CustID, FoodID)

vALUES('01', 'Chuckie', 'German Shephard', '06', '68', '', '01', '02');

INSERT INTO DOG(DogID, DogName, DogBreed, DogAge, DogWeight, DogImage, CustID, FoodID)

VALUES('02', 'Bean', 'Golden Retriever', '04', '77', '', '02', '01');

INSERT INTO DOG(DogID, DogName, DogBreed, DogAge, DogWeight, DogImage, CustID, FoodID)

VALUES('03', 'Tommy', 'Poodle', '03', '55', '', '02', '01');

INSERT INTO DOG(DogID, DogName, DogBreed, DogAge, DogWeight, DogImage, CustID, FoodID)

VALUES('04', 'Ralphy', 'Bulldog', '04', '52', '', '03', '02');

INSERT INTO DOG(DogID, DogName, DogBreed, DogAge, DogWeight, DogImage, CustID, FoodID)

VALUES('05', 'Mickey', 'Dachshund', '07', '28', '', '04', '06');

INSERT INTO DOG(DogID, DogName, DogBreed, DogAge, DogWeight, DogImage, CustID, FoodID)

VALUES('06', 'Draco', 'Siberian Husky', '03', '49', '', '05', '04');

INSERT INTO DOG(DogID, DogName, DogBreed, DogAge, DogWeight, DogImage, CustID, FoodID)

VALUES('07', 'Carlos', 'Chihuahua', '02', '04', '', '05', '04');

INSERT INTO DOG(DogID, DogName, DogBreed, DogAge, DogWeight, DogImage, CustID, FoodID)

VALUES('08', 'Mobi', 'Bulldog', '02', '55', '', '03', '02');

-- PAYMENT(PayID, PayCardNumber, PayExpirationDate, PayType, CustID)

INSERT INTO PAYMENT(PayID, PayCardNumber, PayExpirationDate, PayType, CustID)

VALUES('01', '5667', '31-JUL-23', 'VISA', '05');

INSERT INTO PAYMENT(PayID, PayCardNumber, PayExpirationDate, PayType, CustID)

VALUES('02', '7812', '30-APR-20', 'AMEX', '03');

INSERT INTO PAYMENT(PayID, PayCardNumber, PayExpirationDate, PayType, CustID)

VALUES('03', '9090', '31-DEC-23', 'VISA', '02');

INSERT INTO PAYMENT(PayID, PayCardNumber, PayExpirationDate, PayType, CustID)

VALUES('04', '1778', '31-JAN-24', 'AMEX', '01');

INSERT INTO PAYMENT(PayID, PayCardNumber, PayExpirationDate, PayType, CustID)

VALUES('05', '2334', '01-AUG-23', 'VISA', '04');

-- INVOICE(InvoiceID, InvoiceDate, InvoiceTotalPrice, CustID, FoodID, PayID)

INSERT INTO INVOICE(InvoiceID, InvoiceDate, InvoiceTotalPrice, CustID, FoodID, PayID)

VALUES('01', '12-MAR-21', '15.99', '04', '06', '05');

INSERT INTO INVOICE(InvoiceID, InvoiceDate, InvoiceTotalPrice, CustID, FoodID, PayID)

VALUES('02', '01-FEB-21', '23.99', '02', '01', '03');

INSERT INTO INVOICE(InvoiceID, InvoiceDate, InvoiceTotalPrice, CustID, FoodID, PayID)

VALUES('03', '06-JAN-21', '32.95', '05', '04', '01');

INSERT INTO INVOICE(InvoiceID, InvoiceDate, InvoiceTotalPrice, CustID, FoodID, PayID)

VALUES('04', '23-MAR-21', '26.99', '01', '02', '04');

INSERT INTO INVOICE(InvoiceID, InvoiceDate, InvoiceTotalPrice, CustID, FoodID, PayID)

VALUES('05', '01-MAR-21', '23.99', '02', '01', '03');

INSERT INTO INVOICE(InvoiceID, InvoiceDate, InvoiceTotalPrice, CustID, FoodID, PayID)

VALUES('06', '02-APR-21', '18.99', '03', '02', '02');

INSERT INTO INVOICE(InvoiceID, InvoiceDate, InvoiceTotalPrice, CustID, FoodID, PayID)

VALUES('07', '01-APR-21', '23.99', '02', '01', '03');

INSERT INTO INVOICE(InvoiceID, InvoiceDate, InvoiceTotalPrice, CustID, FoodID, PayID)

VALUES('08', '16-APR-21', '20.99', '05', '05', '01');

-- SHIPMENT(ShipID, ShipRoadNumber, ShipRoadName, ShipCity, ShipState, ShipZip, InvoiceID)

INSERT INTO SHIPMENT(ShipID, ShipRoadNumber, ShipRoadName, ShipCity, ShipState, ShipZip, InvoiceID)

VALUES('01', '943', 'Shiny Lane', 'Fairborn', 'OH', '45324', '01');

INSERT INTO SHIPMENT(ShipID, ShipRoadNumber, ShipRoadName, ShipCity, ShipState, ShipZip, InvoiceID)

VALUES('02', '1567', 'Villa Crossing', 'Fairborn', 'OH', '45324', '02');

INSERT INTO SHIPMENT(ShipID, ShipRoadNumber, ShipRoadName, ShipCity, ShipState, ShipZip, InvoiceID)

VALUES('03', '606', 'Faith Road', 'Dayton', 'OH', '45402', '03');

INSERT INTO SHIPMENT(ShipID, ShipRoadNumber, ShipRoadName, ShipCity, ShipState, ShipZip, InvoiceID)

VALUES('04', '788', 'Square Blvd', 'Beavercreek', 'OH', '45435', '04');

INSERT INTO SHIPMENT(ShipID, ShipRoadNumber, ShipRoadName, ShipCity, ShipState, ShipZip, InvoiceID)

VALUES('05', '1567', 'Villa Crossing', 'Fairborn', 'OH', '45324', '05');

INSERT INTO SHIPMENT(ShipID, ShipRoadNumber, ShipRoadName, ShipCity, ShipState, ShipZip, InvoiceID)

VALUES('06', '216', 'Tree Road', 'Beavercreek', 'OH', '45435', '06');

INSERT INTO SHIPMENT(ShipID, ShipRoadNumber, ShipRoadName, ShipCity, ShipState, ShipZip, InvoiceID)

VALUES('07', '1567', 'Villa Crossing', 'Fairborn', 'OH', '45324', '07');

INSERT INTO SHIPMENT(ShipID, ShipRoadNumber, ShipRoadName, ShipCity, ShipState, ShipZip, InvoiceID)

VALUES('08', '606', 'Faith Road', 'Dayton', 'OH', '45402', '03');

-- Three Table Join with a Restriction

SELECT CustFirst, CustLast, DogName, DogBreed, DogWeight, FoodBrand

FROM CUSTOMER, DOG, DOGFOOD

WHERE CUSTOMER.CustID = DOG.CustID

AND DOG.FoodID = DOGFOOD.FoodID

AND DOG.DogWeight >= 50

ORDER BY DogWeight;

-- Two Table Statistical Join with a Single Row Function

Select DOGFOOD.FOODID, DOGFOOD.FOODBRAND, AVG(INVOICETOTALPRICE) as "Quarterly Costs", COUNT(INVOICE.FOODID) as "Amount of times purchased"

From INVOICE, DOGFOOD

WHERE InvoiceDate >= add\_months(sysdate, -3)

and INVOICE.FOODID = DOGFOOD.FOODID

GROUP BY DOGFOOD.FOODID, DOGFOOD.FOODBRAND

ORDER BY FOODID;

-- Nested SubQuery

-- Display the highest invoice total price and the city name it was shipped to

SELECT ShipCity, InvoiceTotalPrice

FROM INVOICE, SHIPMENT

WHERE INVOICE.InvoiceID = SHIPMENT.InvoiceID

AND SHIPMENT.ShipCity IN

(SELECT ShipCity

FROM SHIPMENT, INVOICE

WHERE INVOICE.InvoiceID = SHIPMENT.InvoiceID

AND TO\_CHAR(INVOICE.InvoiceDate, 'MM') = 3 )

AND INVOICE.InvoiceTotalPrice IN

(SELECT MAX(InvoiceTotalPrice)

FROM INVOICE

WHERE TO\_CHAR(INVOICE.InvoiceDate, 'MM') = 3 );

-- Create Materialized View

-- Display Customer info along with their dog and the dog's favorite food

CREATE MATERIALIZED VIEW vwCustomerDogFavoriteFood

AS SELECT CustFirst as FIRST\_NAME, CustLast as LAST\_NAME, DOGNAME as DOG\_NAME,

DOGFOOD.FOODBRAND as FAVORITE\_FOOD

FROM CUSTOMER, DOG, DOGFOOD

WHERE CUSTOMER.CUSTID = DOG.CUSTID

and DOG.FOODID = DOGFOOD.FOODID

Order by CUSTOMER.CUSTID;

-- Create Materialized View

-- Displays Customer name and their respective invoice with the dog food they purchased

CREATE MATERIALIZED VIEW vwCustomerInvoiceDogFood

AS SELECT CustFirst as FIRST\_NAME, CustLast as LAST\_NAME,

INVOICE.INVOICEDATE AS INVOICE\_DATE, INVOICE.INVOICETOTALPRICE AS PURCHASE\_COSTS,

DOGFOOD.FOODBRAND as PURCHASED\_FOOD

FROM CUSTOMER, INVOICE, DOGFOOD

WHERE CUSTOMER.CUSTID = INVOICE.CUSTID

and INVOICE.FOODID = DOGFOOD.FOODID

ORDER BY INVOICEDATE;

-- PL/SQL Statement

-- Automate increment of dog age every year

DECLARE

CURSOR DogAgeCursor IS

SELECT DogName, DogAge

FROM DOG

WHERE DogAge > 5;

DogName DOG.DogName%type;

NewDogAge DOG.DogAge%type;

BEGIN

--Update dog ages to be one year older

UPDATE DOG

SET DogAge = DogAge + 1;

--Intro messages

DBMS\_OUTPUT.PUT\_LINE('A year has passed!');

DBMS\_OUTPUT.PUT\_LINE('DOGS OVER 5 YEARS OLD:');

OPEN DogAgeCursor;

LOOP

FETCH DogAgeCursor INTO DogName,NewDogAge;

EXIT WHEN DogAgeCursor%NOTFOUND;

DBMS\_OUTPUT.PUT\_LINE(DogName||' is now '||NewDogAge||' years old!');

END LOOP;

CLOSE DogAgeCursor;

END;