ULTIMUTT WALKS

WE PICK UP WHEN YOU CAN'T

THE DOG DATABASE PROJECT

Summary

Ultimutt Walks is a dog walking business located in Fort Lauderdale, FL and has been operating since April of 2019. Currently Ultimutt Walks has no formal system in place to keep track of any data for the business. Their data is currently being kept on several planners, in staff phones, or not stored anywhere at all. We propose to implement a database for Ultimutt Walks so that they may track customers, scheduling, profits, and to identify any trends that may be useful to increase business. In Phase 1 we will be defining the business in terms of entities and relationships by determining business rules, providing a data glossary, identifying the questions we would like to answer as a result of this project, and demonstrating both a conceptual and logical model of the future database. After review and upon approval from the stakeholders involved, we will then be able to move to the next phase of implementing this database.

Stakeholders

The stakeholders included in this project are the owner, the employees, customers, their current investors, and their potential business partner Wag, who has discussed buyouts with Ultimutt Walks. We have proposed that the implementation of a database will increase the profitability of the Company and therefore will affect all stakeholders positively. Customers will be able to access their past and scheduled walks and make updates about their pets if needed. Employees will be able to view their walk schedule and log any information about each walk completed. Investors, owners, and potential business partners will be able to track profitability, number of walks each week, and have an overview of day-to-day business.

Business Rules

- A customer owns one or more pets.
- A customer places an order.
- A staff member completes an order.
- An order contains order details.
- An order status updates the order.
- A dog walk makes up an order detail.
- A pet is included in an order detail.
- A staff payment is paid to a staff.
- An order creates an invoice.
- An invoice payment completes the invoice.
- An invoice status updates the invoice.

Glossary

A **customer** is a person who schedules a service through Ultimutt Walks.

A **pet** is an animal owned by a person.

An **order** is placed by a customer and contains the time the order is placed, which staff member will complete it, the customer who placed the order, and the staff member assigned to the order.

An **order detail** includes the requested date for the service to take place, the amount charged for the order, the status of the order, and may contain details about the dog walk.

A **dog walk** is a service where a pet or pets is taken outside for a certain number of miles or minutes.

The **order status** can either be OPEN or CLOSED.

A **CLOSED order status** means that a staff member has completed the order.

An **OPEN order status** means that an order is scheduled and assigned to a staff member.

A **staff member** is a person employed by Ultimutt Walks.

A **staff payment** is earnings paid to one of the staff after the customer pays for the service.

An **invoice** is sent to the customer to be paid after their order is completed.

An **invoice status** is either PAID or UNPAID.

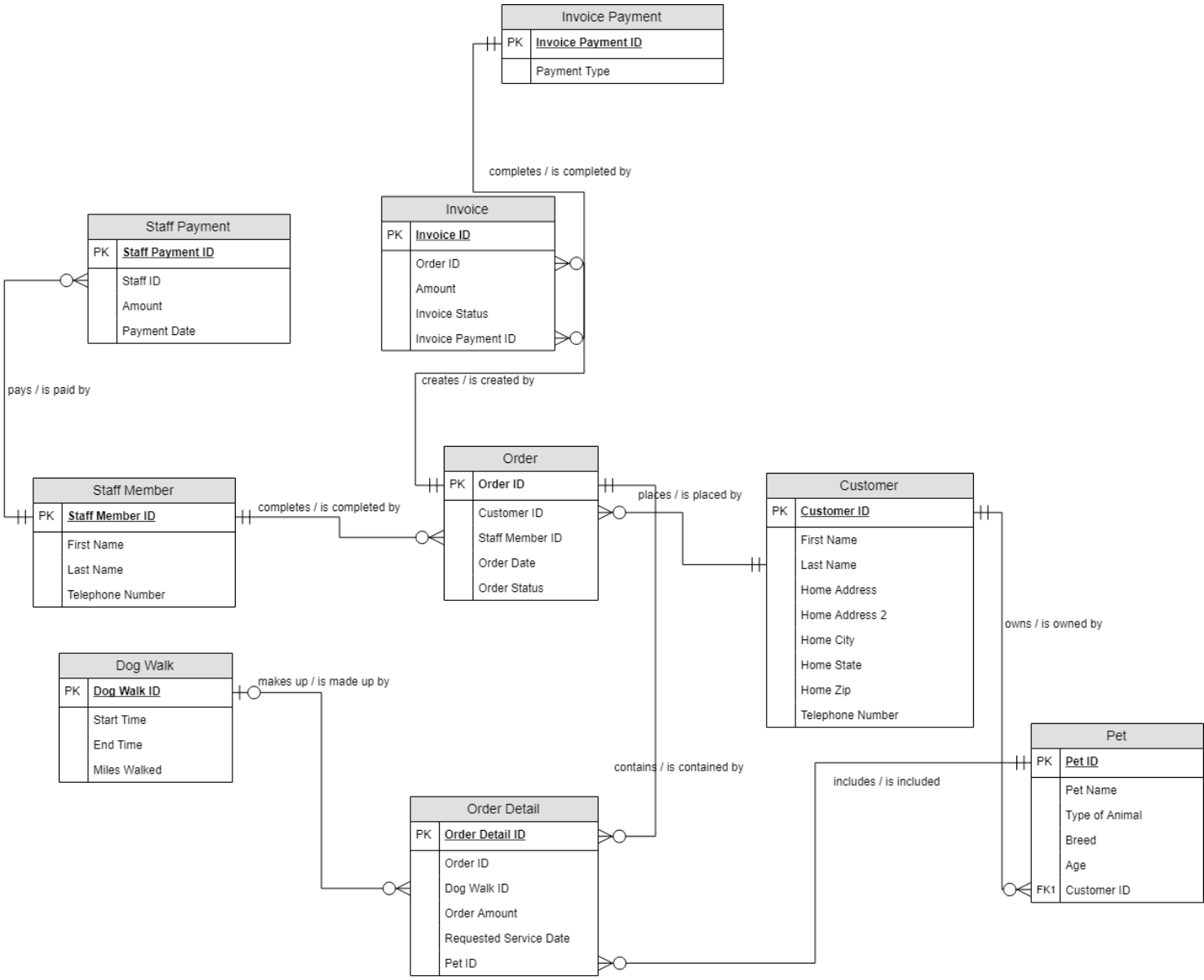
A **PAID invoice status** is when Ultimutt Walks has received payment for a completed order.

An **UNPAID invoice status** is when Ultimutt Walks has not received any payment.

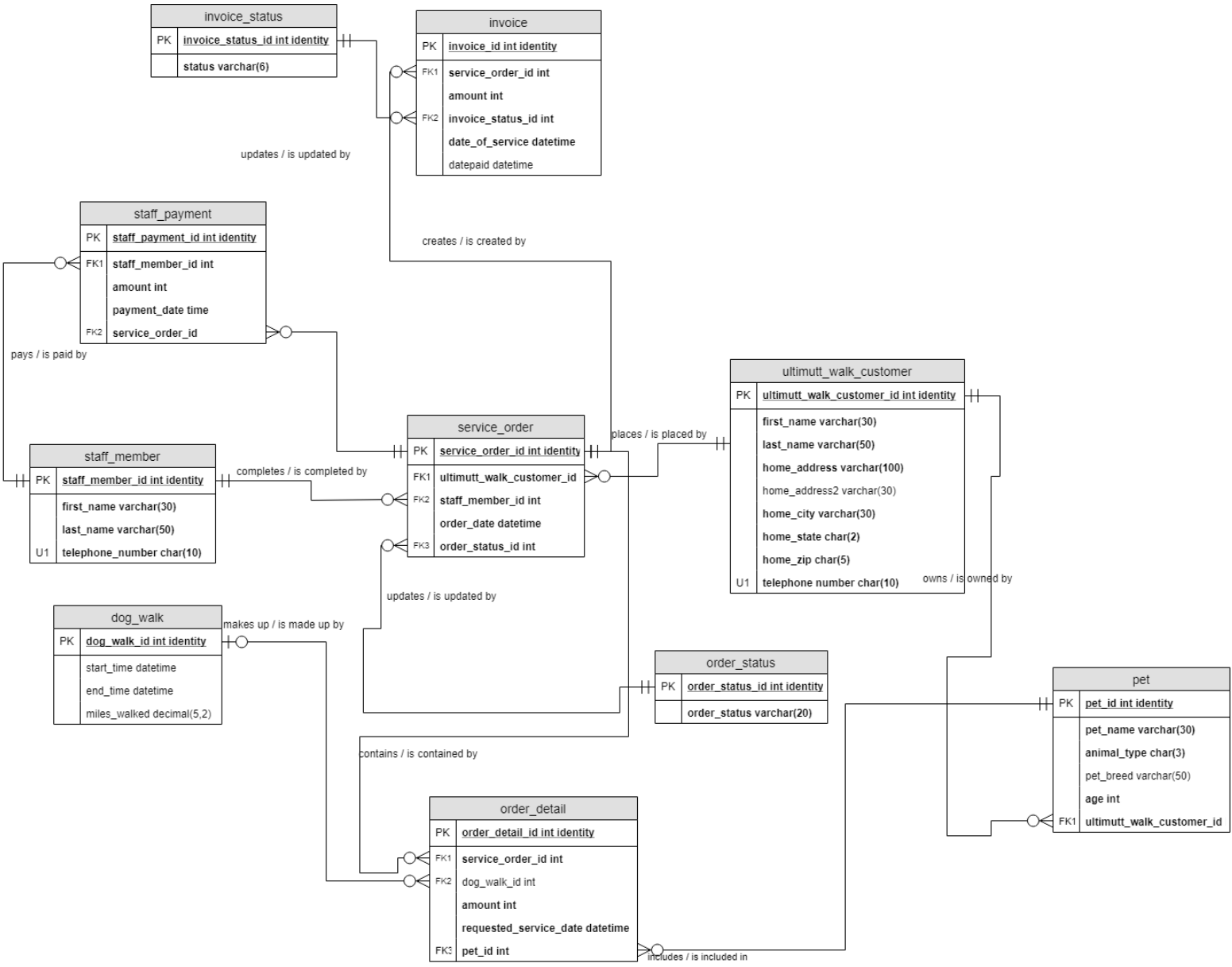
Data Questions

- How many walks are requested for the same day versus being prescheduled?
- What are the most serviced zip codes?
- What are the profits of the company at any given moment?
- What is the average length of a dog walk? How many miles are walked a month?
- What is the average age for a pet and what is the most common breed?

Conceptual Model

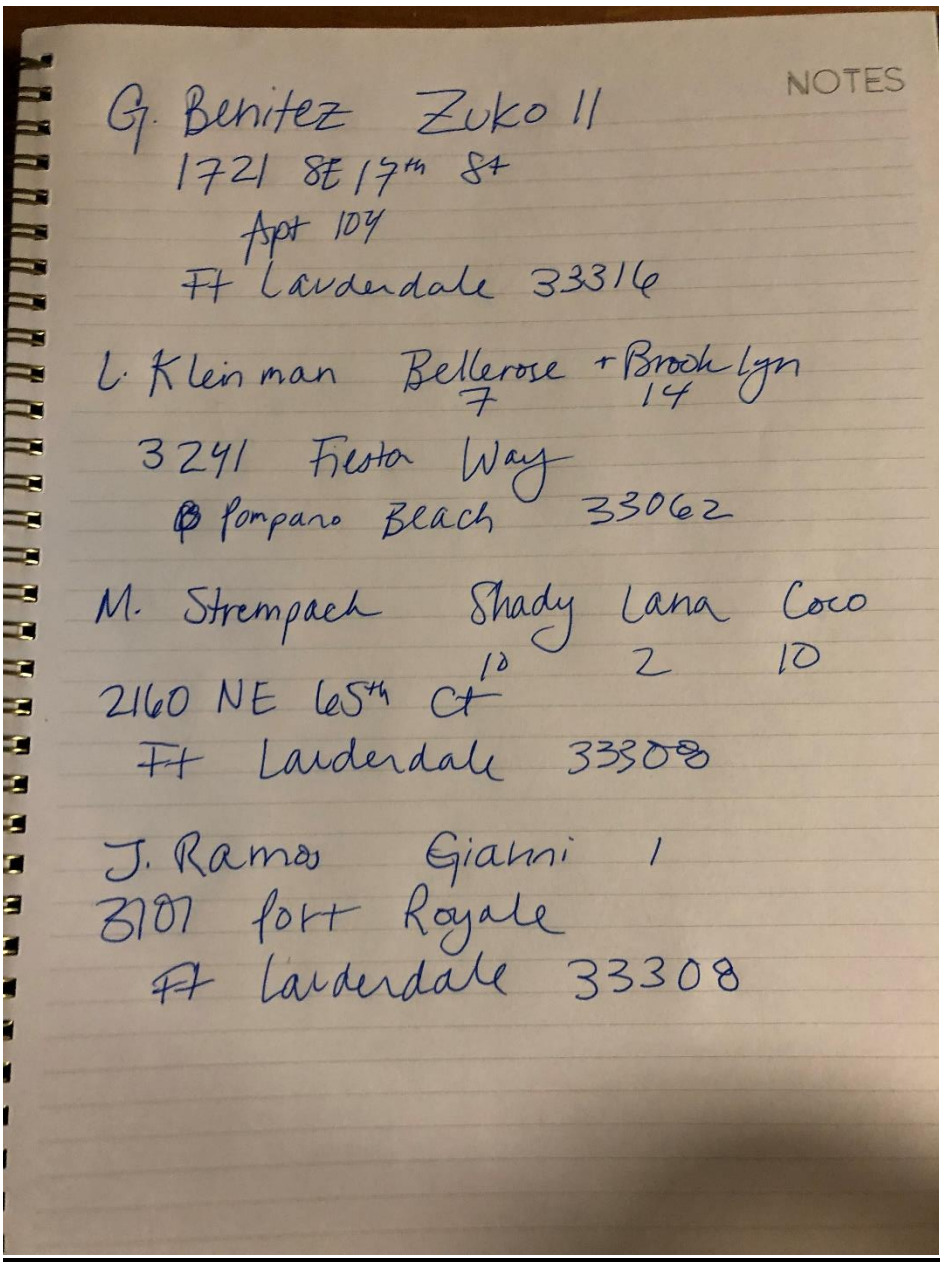


Logical Model



Raw Data Sample:

As aforementioned, Ultimutt Walks previously was using notebooks, phones, and from memory to store data. Below is Customer and Pet information from one of the company's notebooks.



Physical Database Design

```
--create tables/views/procedures/functions in repeatable form
-- create drops

--drop procedures
IF OBJECT_ID('dbo.ViewMyPayments') IS NOT NULL
	DROP PROCEDURE dbo.ViewMyPayments
IF OBJECT_ID('dbo.CloseOutOrder') IS NOT NULL
	DROP PROCEDURE dbo.CloseOutOrder

--drop views
IF OBJECT_ID('dbo.ClosedOrders') IS NOT NULL
	DROP VIEW dbo.ClosedOrders
IF OBJECT_ID('dbo.Invoices') IS NOT NULL
	DROP VIEW dbo.Invoices
IF OBJECT_ID('dbo.OpenOrders') IS NOT NULL
	DROP VIEW dbo.OpenOrders
IF OBJECT_ID('dbo.PetandOwners') IS NOT NULL
DROP VIEW PetandOwners
IF OBJECT_ID('dbo.totalprofit') IS NOT NULL
DROP VIEW dbo.totalprofit
IF OBJECT_ID('dbo.weekly_profit') IS NOT NULL
DROP VIEW dbo.weekly_profit

--drop tables
IF OBJECT_ID('dbo.staff_payment') IS NOT NULL
	DROP TABLE dbo.staff_payment
IF OBJECT_ID('dbo.invoice') IS NOT NULL
	DROP TABLE dbo.invoice
IF OBJECT_ID('invoice_status') IS NOT NULL
	DROP TABLE dbo.invoice_status
IF OBJECT_ID('dbo.order_detail') IS NOT NULL
	DROP TABLE dbo.order_detail
IF OBJECT_ID('dbo.service_order') IS NOT NULL
	DROP TABLE dbo.service_order
IF OBJECT_ID('dbo.order_status') IS NOT NULL
	DROP TABLE dbo.order_status
IF OBJECT_ID('dbo.staff_member') IS NOT NULL
	DROP TABLE dbo.staff_member
IF OBJECT_ID('dbo.dog_walk') IS NOT NULL
	DROP TABLE dbo.dog_walk
IF OBJECT_ID('dbo.pet') IS NOT NULL
	DROP TABLE dbo.pet
IF OBJECT_ID('ultimutt_walk_customer') IS NOT NULL
	DROP TABLE dbo.ultimutt_walk_customer

--create database tables
CREATE TABLE ultimutt_walk_customer(
	--create columns
	ultimutt_walk_customer_id int identity
	, first_name varchar(50) NOT NULL
	, last_name varchar(50) NOT NULL
	, home_address varchar(200) NOT NULL
	, home_address2 varchar(50)
```

```
	, home_city varchar(30) NOT NULL
	, home_state char(3) NOT NULL
	, home_zip char(5) NOT NULL
	, telephone_number char(10) NOT NULL
	--place constraints
	, CONSTRAINT PK_ultimutt_walk_customer PRIMARY KEY (ultimutt_walk_customer_id)
	, CONSTRAINT U1_ultimutt_walk_customer UNIQUE(telephone_number)
	)

CREATE TABLE pet(
	--create columns
	pet_id int identity
	, pet_name varchar(30) NOT NULL
	, animal_type char(3) NOT NULL
	, pet_breed varchar(50) NOT NULL
	, age int NOT NULL
	, ultimutt_walk_customer_id int NOT NULL
	--place constraints
	, CONSTRAINT PK_pet PRIMARY KEY(pet_id)
	, CONSTRAINT FK1_pet FOREIGN KEY(ultimutt_walk_customer_id) REFERENCES
ultimutt_walk_customer(ultimutt_walk_customer_id)
	)

CREATE TABLE dog_walk(
	--create columns
	dog_walk_id int identity
	, start_time datetime
	, end_time datetime
	, miles_walked decimal(5,2)
	-- place constraints
	, CONSTRAINT PK_dog_walk PRIMARY KEY(dog_walk_id)
	)


CREATE TABLE staff_member(
	--create columns
	staff_member_id int identity
	, first_name varchar(30)NOT NULL
	, last_name varchar(50) NOT NULL
	, telephone_number char(10) NOT NULL
	--place constraints
	, CONSTRAINT PK_staff_member PRIMARY KEY (staff_member_id)
	, CONSTRAINT U1_staff_member UNIQUE(telephone_number)
	)


CREATE TABLE order_status(
	--create columns
	order_status_id int identity
	, order_status varchar(20) NOT NULL
	--place constraints
	, CONSTRAINT PK_order_status PRIMARY KEY (order_status_id)
	)

CREATE TABLE service_order(
	--create columns
	service_order_id int identity
	, ultimutt_walk_customer_id int NOT NULL
```

```
	, staff_member_id int NOT NULL
	, order_date datetime NOT NULL
	, order_status_id int NOT NULL
	--place constraints
	, CONSTRAINT PK_service_order PRIMARY KEY(service_order_id)
	, CONSTRAINT FK1_service_order FOREIGN KEY(ultimutt_walk_customer_id) REFERENCES
ultimutt_walk_customer(ultimutt_walk_customer_id)
	, CONSTRAINT FK2_service_order FOREIGN KEY(staff_member_id) REFERENCES
staff_member(staff_member_id)
	, CONSTRAINT FK3_service_order FOREIGN KEY(order_status_id) REFERENCES
order_status(order_status_id)
	)


CREATE TABLE order_detail(
	--create columns
	order_detail_id int identity
	, service_order_id int NOT NULL
	, dog_walk_id int
	, amount int NOT NULL
	, requested_service_date datetime NOT NULL
	, pet_id int NOT NULL
	--place constraints
	, CONSTRAINT PK_order_detail PRIMARY KEY(order_detail_id)
	, CONSTRAINT FK1_order_detail FOREIGN KEY(service_order_id) REFERENCES
service_order(service_order_id)
	, CONSTRAINT FK2_order_detail FOREIGN KEY(dog_walk_id) REFERENCES
dog_walk(dog_walk_id)
	, CONSTRAINT FK3_order_detail FOREIGN KEY(pet_id) REFERENCES pet(pet_id)
	)



CREATE TABLE invoice_status(
	--create columns
	invoice_status_id int identity
	, invoice_status varchar(6) NOT NULL
	--place constraints
	, CONSTRAINT PK_invoice_status PRIMARY KEY(invoice_status_id)
	)

CREATE TABLE invoice(
	--create columns
	invoice_id int identity
	, service_order_id int NOT NULL
	, amount int NOT NULL
	, invoice_status_id int NOT NULL
	, date_of_service datetime NOT NULL
	, datepaid datetime
	--place constraints
	, CONSTRAINT PK_invoice PRIMARY KEY(invoice_id)
	, CONSTRAINT FK1_invoice FOREIGN KEY(service_order_id) REFERENCES
service_order(service_order_id)
	, CONSTRAINT FK2_invoice FOREIGN KEY(invoice_status_id) REFERENCES
invoice_status(invoice_status_id)
	)

CREATE TABLE staff_payment(
```

```
	-- create column
	staff_payment_id int identity
	, staff_member_id int
	, amount int NOT NULL
	, payment_date datetime NOT NULL
	, service_order_id int NOT NULL
	--place constraints
	, CONSTRAINT PK_staff_payment PRIMARY KEY(staff_payment_id)
	, CONSTRAINT FK1_staff_payment FOREIGN KEY(staff_member_id) REFERENCES
staff_member(staff_member_id)
	, CONSTRAINT FK2_staff_payment FOREIGN KEY(service_order_id) REFERENCES
service_order(service_order_id)
	)
--insert allcustomers into ultimutt_walk_customer table
--NOTE Barbara Fehrenbach is moving, will need updated address
INSERT INTO ultimutt_walk_customer(first_name, last_name, home_address, home_address2,
home_city, home_state, home_zip, telephone_number)
VALUES ('Jose', 'Ramos', '3101 Port Royale Blvd', 'Apt 636', 'Fort Lauderdale', 'FL',
'33308', '9542940388')
, ('Mimi', 'Strempack', '2160 NE 65th Ct', NULL, 'Fort Lauderdale', 'FL', '33308',
'9542950685')
, ('Lily', 'Kleinman', '3241 Fiesta Way', NULL, 'Pompano Beach', 'FL', '33062',
'5612134947')
, ('Brady', 'Bunch', '1371 NE 27th Way', NULL, 'Pompano Beach', 'FL', '33062',
'7273658881')
, ('Cher', 'Brahmer', '611 SE 9th Ave', NULL, 'Pompano Beach', 'FL', '33060',
'9544489222')
, ('Lady', 'Cooke', '918 NE 17th Terrace', 'Apt 1', 'Fort Lauderdale', 'FL', '33304',
'8602480981')
, ('Michael', 'London', '611 SW 11th St', 'Apt 2', 'Fort Lauderdale', 'FL', '33315',
'9546085634')
, ('Jack', 'Scott', '1626 SE 1st St', NULL, 'Fort Lauderdale', 'FL', '33301',
'9548951400')
, ('Demi', 'Lovato', '2121 S Ocean Blvd', 'Unit 303', 'Pompano Beach', 'FL', '33062',
'7046617161')
, ('Talula', 'Wright', '2137 N Cypress Bend Dr', 'Apt 404', 'Pompano Beach', 'FL',
'33069', '9542708435')
, ('Scooby', 'Doo', '3101 Port Royale Blvd', 'Apt 534', 'Fort Lauderdale', 'FL', '33308',
'5615421689')
, ('Pooh', 'Bear', '1620 SE 2nd CT', NULL, 'Fort Lauderdale', 'FL', '33301',
'3152444296')
, ('Greg', 'Benitez','1721 SE 17th St' , '104', 'Fort Lauderdale', 'FL', '33316',
'9547328587')
, ('Barbs', 'Fehrenback', 'John Knox Village', NULL, 'Pompano Beach', 'FL', '33062',
'6317664449')
, ('Shania', 'Twain', 'Unknown', NULL, 'Oakland Park', 'FL', '33305', '4077566083')
, ('Ted', 'Rochester', '3101 Port Royale Blvd', NULL, 'Fort Lauderdale', 'FL', '33308',
'9547893293')

--look at inserts to check for mistakes
SELECT * FROM ultimutt_walk_customer
```

	ultimutt_walk_customer_id	first_name	last_name	home_address	home_address2	home_city	home_state	home_zip	telephone_number
1	1	Jose	Ramos	3101 Port Royale Blvd	Apt 636	Fort Lauderdale	FL	33308	9542940388
2	2	Mimi	Strempack	2160 NE 65th Ct	NULL	Fort Lauderdale	FL	33308	9542950685
3	3	Lily	Kleinman	3241 Fiesta Way	NULL	Pompano Beach	FL	33062	5612134947
4	4	Brady	Bunch	1371 NE 27th Way	NULL	Pompano Beach	FL	33062	7273658881
5	5	Cher	Brahmer	611 SE 9th Ave	NULL	Pompano Beach	FL	33060	9544489222
6	6	Lady	Cooke	918 NE 17th Terrace	Apt 1	Fort Lauderdale	FL	33304	8602480981
7	7	Michael	London	611 SW 11th St	Apt 2	Fort Lauderdale	FL	33315	9546085634
8	8	Jack	Scott	1626 SE 1st St	NULL	Fort Lauderdale	FL	33301	9548951400
9	9	Demi	Lovato	2121 S Ocean Blvd	Unit 303	Pompano Beach	FL	33062	7046617161
10	10	Talula	Wright	2137 N Cypress Bend Dr	Apt 404	Pompano Beach	FL	33069	9542708435
11	11	Scooby	Doo	3101 Port Royale Blvd	Apt 534	Fort Lauderdale	FL	33308	5615421689
12	12	Pooh	Bear	1620 SE 2nd CT	NULL	Fort Lauderdale	FL	33301	3152444296
13	13	Greg	Benitez	1721 SE 17th St	104	Fort Lauderdale	FL	33316	9547328587
14	14	Barbs	Fehrenb...	John Knox Village	NULL	Pompano Beach	FL	33062	6317664449
15	15	Shania	Twain	Unknown	NULL	Oakland Park	FL	33305	4077566083
16	16	Ted	Rochester	3101 Port Royale Blvd	NULL	Fort Lauderdale	FL	33308	9547893293

```
--16 solid customers so far! that's great

--insert pet info, identify customer ID through unique telephone number provided
INSERT INTO pet(pet_name, animal_type, pet_breed, age, ultimutt_walk_customer_id)
VALUES
       ('Gianni', 'dog', 'French Bulldog', 1, (SELECT ultimutt_walk_customer_id FROM
ultimutt_walk_customer WHERE telephone_number = '9542940388'))
       , ('Coco', 'dog', 'Dachshund', 8, (SELECT ultimutt_walk_customer_id FROM
ultimutt_walk_customer WHERE telephone_number = '9542950685'))
       , ('Shady', 'dog', 'Dachshund', 8, (SELECT ultimutt_walk_customer_id FROM
ultimutt_walk_customer WHERE telephone_number = '9542950685'))
       , ('Lana', 'dog', 'Shih Tzu', 8, (SELECT ultimutt_walk_customer_id FROM
ultimutt_walk_customer WHERE telephone_number = '9542950685'))
       , ('Bellerose', 'dog', 'Min Pin', 8, (SELECT ultimutt_walk_customer_id FROM
ultimutt_walk_customer WHERE telephone_number = '5612134947'))
       , ('Brooklyn', 'dog', 'Papillon', 14, (SELECT ultimutt_walk_customer_id FROM
ultimutt_walk_customer WHERE telephone_number = '5612134947'))
       , ('Charlie', 'cat', 'Orange Cat', 14, (SELECT ultimutt_walk_customer_id FROM
ultimutt_walk_customer WHERE telephone_number ='7273658881'))
       , ('Lil Bit', 'cat', 'White Cat', 3, (SELECT ultimutt_walk_customer_id FROM
ultimutt_walk_customer WHERE telephone_number ='9544489222'))
       , ('Tux', 'cat', 'Black & White Cat', 3, (SELECT ultimutt_walk_customer_id FROM
ultimutt_walk_customer WHERE telephone_number ='9544489222'))
       , ('Checko', 'dog', 'Mutt - Maybe Australian Shepherd', 5, (SELECT
ultimutt_walk_customer_id FROM ultimutt_walk_customer WHERE telephone_number
='8602480981'))
       , ('Barlo', 'dog', 'Pitbull & Black Mouth Curr', 1, (SELECT
ultimutt_walk_customer_id FROM ultimutt_walk_customer WHERE telephone_number =
'9546085634'))
       , ('Opi', 'dog', 'Italian Greyhound', 4, (SELECT ultimutt_walk_customer_id FROM
ultimutt_walk_customer WHERE telephone_number = '9548951400'))
       , ('Boo', 'dog', 'Italian Greyhound', 3, (SELECT ultimutt_walk_customer_id FROM
ultimutt_walk_customer WHERE telephone_number = '9548951400'))
       , ('Barney', 'dog', 'King Charles Cavalier', 12, (SELECT ultimutt_walk_customer_id
FROM ultimutt_walk_customer WHERE telephone_number = '7046617161'))
       , ('Mocha', 'dog', 'Mini Schnauzer', 3, (SELECT ultimutt_walk_customer_id FROM
ultimutt_walk_customer WHERE telephone_number = '9542708435'))
       , ('Shadow', 'dog', 'Husky', 4, (SELECT ultimutt_walk_customer_id FROM
ultimutt_walk_customer WHERE telephone_number = '5615421689'))
       , ('Sunny', 'dog', 'Great Dane & Lab Mix', 7, (SELECT ultimutt_walk_customer_id
FROM ultimutt_walk_customer WHERE telephone_number = '3152444296'))
```

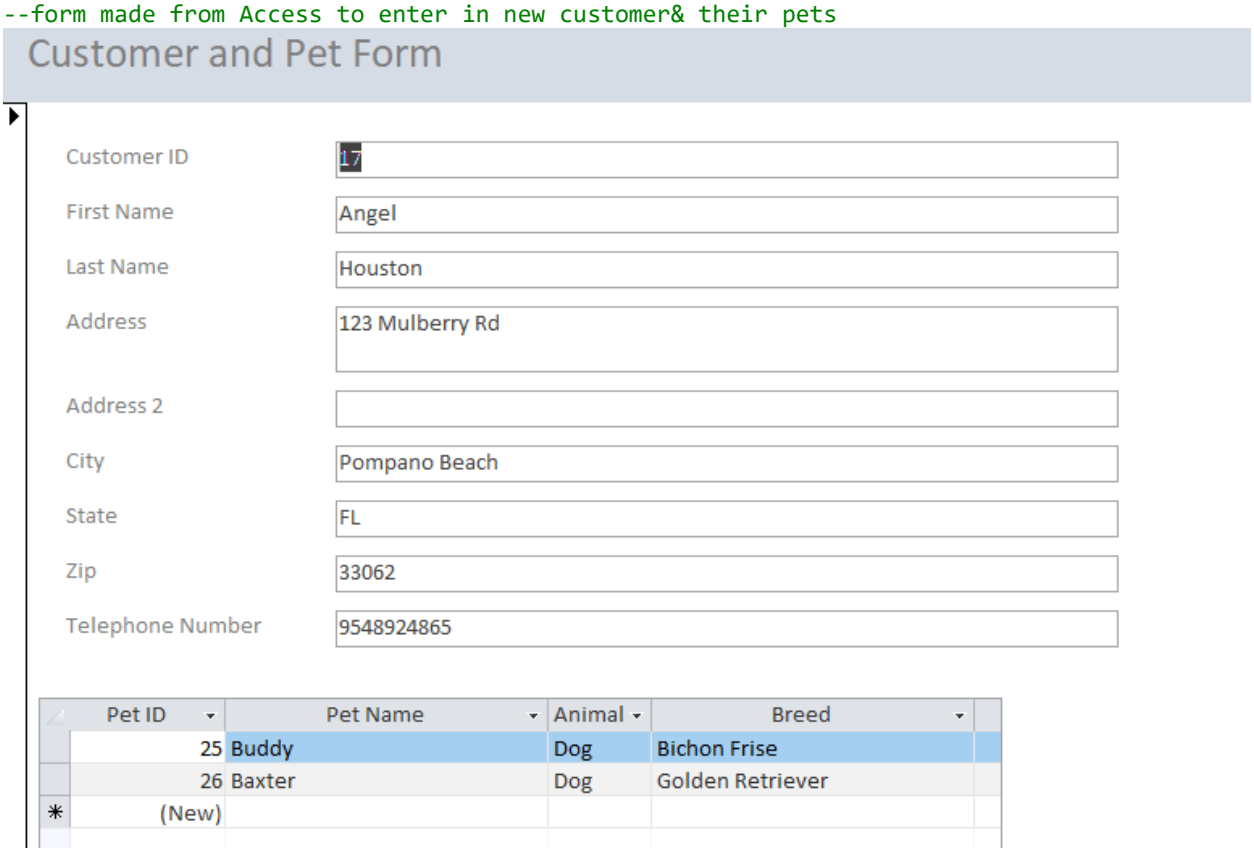
```
       , ('Ruby', 'dog', 'Pitbull & Lab Mix', 7, (SELECT ultimutt_walk_customer_id FROM
ultimutt_walk_customer WHERE telephone_number = '3152444296'))
       , ('Zuko', 'dog', 'English Bulldog', 8, (SELECT ultimutt_walk_customer_id FROM
ultimutt_walk_customer WHERE telephone_number = '9547328587'))
       , ('Evy', 'dog', 'Dachshund Mix', 4, (SELECT ultimutt_walk_customer_id FROM
ultimutt_walk_customer WHERE telephone_number = '4077566083'))
       , ('Kakao', 'dog', 'Great Dane & Lab Mix', 9, (SELECT ultimutt_walk_customer_id
FROM ultimutt_walk_customer WHERE telephone_number = '4077566083'))
       , ('Sylvie', 'dog', 'Mini Poodle', 3, (SELECT ultimutt_walk_customer_id FROM
ultimutt_walk_customer WHERE telephone_number = '6317664449'))
       , ('Jax', 'dog', 'Puggle', 4, (SELECT ultimutt_walk_customer_id FROM
ultimutt_walk_customer WHERE telephone_number = '9547893293'))
       , ('Gnocchi', 'dog', 'Terrier Mix', 1, (SELECT ultimutt_walk_customer_id FROM
ultimutt_walk_customer WHERE telephone_number = '9547893293'))

--create view of pets and their owners
--NOTE: Later make a form to enter new customers with pets, will be much easier
GO
CREATE VIEW PetandOwners
AS
SELECT
       pet.pet_name as PetName,
       pet.pet_breed as Breed,
       ultimutt_walk_customer.first_name + ' ' + ultimutt_walk_customer.last_name as
Pet_Owner
FROM pet
INNER JOIN ultimutt_walk_customer ON pet.ultimutt_walk_customer_id =
ultimutt_walk_customer.ultimutt_walk_customer_id
GO
SELECT*FROM PetandOwners
```

Results | Messages

	PetName	Breed	Pet_Owner
1	Gianni	French Bulldog	Jose Ramos
2	Coco	Dachshund	Mimi Strempack
3	Shady	Dachshund	Mimi Strempack
4	Lana	Shih Tzu	Mimi Strempack
5	Bellerose	Min Pin	Lily Kleinman
6	Brooklyn	Papillon	Lily Kleinman
7	Charlie	Orange Cat	Brady Bunch
8	Lil Bit	White Cat	Cher Brahmer
9	Tux	Black & Whit...	Cher Brahmer
10	Checko	Mutt - Maybe...	Lady Cooke
11	Barlo	Pitbull & Blac...	Michael London
12	Opi	Italian Greyh...	Jack Scott
13	Boo	Italian Greyh...	Jack Scott
14	Barney	King Charles ...	Demi Lovato
15	Mocha	Mini Schnau...	Talula Wright
16	Shadow	Husky	Scooby Doo
17	Sunny	Great Dane ...	Pooh Bear
18	Ruby	Pitbull & Lab ...	Pooh Bear
19	Zuko	English Bulldog	Greg Benitez
20	Evy	Dachshund ...	Shania Twain
21	Kakao	Great Dane ...	Shania Twain
22	Sylvie	Mini Poodle	Barbs Fehren...
23	Jax	Puggle	Ted Rochester
24	Gnocchi	Terrier Mix	Ted Rochester

```
GO
```

```
--form made from Access to enter in new customer& their pets
```




```
--time to employ staff members!

INSERT INTO staff_member(first_name, last_name, telephone_number)
VALUES ('Katie', 'Hanks', '7867792553')
       , ('Gary', 'Guinta', '6035455470')
       , ('Fran', 'Guinta', '6316262029')

--let's check out the newest members of Ultimutt Walks
SELECT * FROM staff_member

--Create order status options
--Keeping it to OPEN or CLOSED
--OPEN order means that the walk has not been completed
--CLOSED order means that the walk has been finished
--Order Status does NOT reflect invoice status

INSERT INTO order_status(order_status)
VALUES ('OPEN'), ('CLOSED')
--identify ID numbers for open and closed for future orders
SELECT * FROM order_status
--OPEN = 1
--CLOSED = 2

--create invoice status choices: UNPAID or PAID

INSERT INTO invoice_status(invoice_status)
VALUES ('UNPAID'), ('PAID')
--identify ID numbers for UNPAID and PAID
SELECT * FROM invoice_status
```

```sql
--UNPAID = 1
--PAID = 2

-- enter orders for the week of 9/2
--
INSERT INTO service_order(ultimutt_walk_customer_id, order_status_id, staff_member_id,
order_date)
VALUES ((SELECT ultimutt_walk_customer_id FROM ultimutt_walk_customer WHERE
telephone_number= '9542940388'), 1, (SELECT staff_member_id FROM staff_member WHERE
first_name = 'Katie'), '9/4/2019')
       , ((SELECT ultimutt_walk_customer_id FROM ultimutt_walk_customer WHERE
telephone_number= '9542940388'), 1, (SELECT staff_member_id FROM staff_member WHERE
first_name = 'Katie'), '9/4/2019')
       , ((SELECT ultimutt_walk_customer_id FROM ultimutt_walk_customer WHERE
telephone_number= '9547328587'), 1, (SELECT staff_member_id FROM staff_member WHERE
first_name = 'Katie'), '9/4/2019')
       , ((SELECT ultimutt_walk_customer_id FROM ultimutt_walk_customer WHERE
telephone_number= '9547328587'), 1, (SELECT staff_member_id FROM staff_member WHERE
first_name = 'Katie'), '9/4/2019')
       , ((SELECT ultimutt_walk_customer_id FROM ultimutt_walk_customer WHERE
telephone_number= '9547328587'), 1, (SELECT staff_member_id FROM staff_member WHERE
first_name = 'Katie'), '9/4/2019')
       , ((SELECT ultimutt_walk_customer_id FROM ultimutt_walk_customer WHERE
telephone_number= '9547328587'), 1, (SELECT staff_member_id FROM staff_member WHERE
first_name = 'Katie'), '9/4/2019')
       , ((SELECT ultimutt_walk_customer_id FROM ultimutt_walk_customer WHERE
telephone_number= '5612134947'), 1, (SELECT staff_member_id FROM staff_member WHERE
first_name = 'Katie'), '9/4/2019')
       , ((SELECT ultimutt_walk_customer_id FROM ultimutt_walk_customer WHERE
telephone_number= '5612134947'), 1, (SELECT staff_member_id FROM staff_member WHERE
first_name = 'Katie'), '9/4/2019')
       , ((SELECT ultimutt_walk_customer_id FROM ultimutt_walk_customer WHERE
telephone_number= '5612134947'), 1, (SELECT staff_member_id FROM staff_member WHERE
first_name = 'Katie'), '9/4/2019')
       , ((SELECT ultimutt_walk_customer_id FROM ultimutt_walk_customer WHERE
telephone_number= '5612134947'), 1, (SELECT staff_member_id FROM staff_member WHERE
first_name = 'Katie'), '9/4/2019')
       , ((SELECT ultimutt_walk_customer_id FROM ultimutt_walk_customer WHERE
telephone_number= '5612134947'), 1, (SELECT staff_member_id FROM staff_member WHERE
first_name = 'Katie'), '9/4/2019')
       , ((SELECT ultimutt_walk_customer_id FROM ultimutt_walk_customer WHERE
telephone_number= '7046617161'), 1, (SELECT staff_member_id FROM staff_member WHERE
first_name = 'Katie'), '9/4/2019')
       , ((SELECT ultimutt_walk_customer_id FROM ultimutt_walk_customer WHERE
telephone_number= '7046617161'), 1, (SELECT staff_member_id FROM staff_member WHERE
first_name = 'Katie'), '9/4/2019')
       , ((SELECT ultimutt_walk_customer_id FROM ultimutt_walk_customer WHERE
telephone_number= '7046617161'), 1, (SELECT staff_member_id FROM staff_member WHERE
first_name = 'Katie'), '9/4/2019')
       , ((SELECT ultimutt_walk_customer_id FROM ultimutt_walk_customer WHERE
telephone_number= '7046617161'), 1, (SELECT staff_member_id FROM staff_member WHERE
first_name = 'Katie'), '9/4/2019')
       , ((SELECT ultimutt_walk_customer_id FROM ultimutt_walk_customer WHERE
telephone_number= '5612134947'), 1, (SELECT staff_member_id FROM staff_member WHERE
first_name = 'Katie'), '9/4/2019')
       , ((SELECT ultimutt_walk_customer_id FROM ultimutt_walk_customer WHERE
telephone_number= '5612134947'), 1, (SELECT staff_member_id FROM staff_member WHERE
first_name = 'Katie'), '9/4/2019')
```

```
       , ((SELECT ultimutt_walk_customer_id FROM ultimutt_walk_customer WHERE
telephone_number= '5612134947'), 1, (SELECT staff_member_id FROM staff_member WHERE
first_name = 'Katie'), '9/4/2019')
       , ((SELECT ultimutt_walk_customer_id FROM ultimutt_walk_customer WHERE
telephone_number= '5612134947'), 1, (SELECT staff_member_id FROM staff_member WHERE
first_name = 'Katie'), '9/4/2019')
       , ((SELECT ultimutt_walk_customer_id FROM ultimutt_walk_customer WHERE
telephone_number= '5612134947'), 1, (SELECT staff_member_id FROM staff_member WHERE
first_name = 'Katie'), '9/4/2019')


--now let's get into the details of next week's orders
INSERT INTO order_detail(service_order_id, requested_service_date, pet_id, amount)
VALUES (1, '9/5/2019', (SELECT pet_id FROM pet WHERE pet_name = 'Gianni'), 15)
       , (2, '9/6/2019', (SELECT pet_id FROM pet WHERE pet_name = 'Gianni'), 15)
       , (3, '9/3/2019', (SELECT pet_id FROM pet WHERE pet_name = 'Zuko'), 15)
       , (4, '9/4/2019', (SELECT pet_id FROM pet WHERE pet_name = 'Zuko'), 15)
       , (5, '9/5/2019', (SELECT pet_id FROM pet WHERE pet_name = 'Zuko'), 15)
       , (6, '9/6/2019', (SELECT pet_id FROM pet WHERE pet_name = 'Zuko'), 15)
       , (7, '9/3/2019', (SELECT pet_id FROM pet WHERE pet_name = 'Bellerose'), 6)
       , (8, '9/4/2019', (SELECT pet_id FROM pet WHERE pet_name = 'Bellerose'), 6)
       , (9, '9/5/2019', (SELECT pet_id FROM pet WHERE pet_name = 'Bellerose'), 6)
       , (10, '9/6/2019', (SELECT pet_id FROM pet WHERE pet_name = 'Bellerose'), 6)
       , (11, '9/7/2019', (SELECT pet_id FROM pet WHERE pet_name = 'Bellerose'), 6)
       , (12, '9/3/2019', (SELECT pet_id FROM pet WHERE pet_name = 'Barney'), 15)
       , (13, '9/4/2019', (SELECT pet_id FROM pet WHERE pet_name = 'Barney'), 15)
       , (14, '9/5/2019', (SELECT pet_id FROM pet WHERE pet_name = 'Barney'), 15)
       , (15, '9/6/2019', (SELECT pet_id FROM pet WHERE pet_name = 'Barney'), 15)
       , (16, '9/3/2019', (SELECT pet_id FROM pet WHERE pet_name = 'Brooklyn'), 6)
       , (17, '9/4/2019 ', (SELECT pet_id FROM pet WHERE pet_name = 'Brooklyn'), 6)
       , (18, '9/5/2019', (SELECT pet_id FROM pet WHERE pet_name = 'Brooklyn'), 6)
       , (19, '9/6/2019', (SELECT pet_id FROM pet WHERE pet_name = 'Brooklyn'), 6)
       , (20, '9/7/2019', (SELECT pet_id FROM pet WHERE pet_name = 'Brooklyn'), 6)


--create a view to see OpenOrders so that order_detail and service_order can be viewed in
a meaningful way
GO
CREATE VIEW OpenOrders
AS
SELECT
       order_detail.order_detail_id as OrderNumber
       , service_order.order_date as OrderDate
       , order_detail.requested_service_date as ServiceDate
       , ultimutt_walk_customer.first_name + ' ' + ultimutt_walk_customer.last_name as
CustomerName
       , pet.pet_name as Pet
       , staff_member.first_name + ' ' + staff_member.last_name as StaffMemberAssigned
       , order_detail.amount as AmountCharged
       , order_status.order_status as OrderStatus
FROM order_detail
INNER JOIN service_order on service_order.service_order_id =
order_detail.service_order_id
INNER JOIN ultimutt_walk_customer on ultimutt_walk_customer.ultimutt_walk_customer_id =
service_order.ultimutt_walk_customer_id
INNER JOIN staff_member ON staff_member.staff_member_id = service_order.staff_member_id
INNER JOIN pet ON pet.pet_id = order_detail.pet_id
INNER JOIN order_status ON order_status.order_status_id=service_order.order_status_id
```

```
WHERE order_status.order_status_id = 1

GO

--Now we can VIEW all open Orders. Will be helpful for my employees Fran and Gary.
SELECT * FROM OpenOrders
```

	OrderNumber	OrderDate	ServiceDate	CustomerName	Pet	StaffMemberAssigned	AmountCharged	OrderStatus
1	1	2019-09-04 00:00:00.000	2019-09-05 00:00:00.000	Jose Ramos	Gianni	Katie Hanks	15	OPEN
2	2	2019-09-04 00:00:00.000	2019-09-06 00:00:00.000	Jose Ramos	Gianni	Katie Hanks	15	OPEN
3	3	2019-09-04 00:00:00.000	2019-09-03 00:00:00.000	Greg Benitez	Zuko	Katie Hanks	15	OPEN
4	4	2019-09-04 00:00:00.000	2019-09-04 00:00:00.000	Greg Benitez	Zuko	Katie Hanks	15	OPEN
5	5	2019-09-04 00:00:00.000	2019-09-05 00:00:00.000	Greg Benitez	Zuko	Katie Hanks	15	OPEN
6	6	2019-09-04 00:00:00.000	2019-09-06 00:00:00.000	Greg Benitez	Zuko	Katie Hanks	15	OPEN
7	7	2019-09-04 00:00:00.000	2019-09-03 00:00:00.000	Lily Kleinman	Bell...	Katie Hanks	6	OPEN
8	8	2019-09-04 00:00:00.000	2019-09-04 00:00:00.000	Lily Kleinman	Bell...	Katie Hanks	6	OPEN
9	9	2019-09-04 00:00:00.000	2019-09-05 00:00:00.000	Lily Kleinman	Bell...	Katie Hanks	6	OPEN
10	10	2019-09-04 00:00:00.000	2019-09-06 00:00:00.000	Lily Kleinman	Bell...	Katie Hanks	6	OPEN
11	11	2019-09-04 00:00:00.000	2019-09-07 00:00:00.000	Lily Kleinman	Bell...	Katie Hanks	6	OPEN
12	12	2019-09-04 00:00:00.000	2019-09-03 00:00:00.000	Demi Lovato	Bam...	Katie Hanks	15	OPEN
13	13	2019-09-04 00:00:00.000	2019-09-04 00:00:00.000	Demi Lovato	Bam...	Katie Hanks	15	OPEN
14	14	2019-09-04 00:00:00.000	2019-09-05 00:00:00.000	Demi Lovato	Bam...	Katie Hanks	15	OPEN
15	15	2019-09-04 00:00:00.000	2019-09-06 00:00:00.000	Demi Lovato	Bam...	Katie Hanks	15	OPEN
16	16	2019-09-04 00:00:00.000	2019-09-03 00:00:00.000	Lily Kleinman	Broo...	Katie Hanks	6	OPEN
17	17	2019-09-04 00:00:00.000	2019-09-04 00:00:00.000	Lily Kleinman	Broo...	Katie Hanks	6	OPEN
18	18	2019-09-04 00:00:00.000	2019-09-05 00:00:00.000	Lily Kleinman	Broo...	Katie Hanks	6	OPEN

```
--View for all CLOSED orders as well - helpful for invoicing
GO
CREATE VIEW ClosedOrders
AS
SELECT
       order_detail.order_detail_id as OrderNumber
       , service_order.order_date as OrderDate
       , order_detail.requested_service_date as ServiceDate
       , ultimutt_walk_customer.first_name + ' ' + ultimutt_walk_customer.last_name as
CustomerName
       , pet.pet_name as Pet
       , staff_member.first_name + ' ' + staff_member.last_name as StaffMemberAssigned
       , order_detail.amount as AmountCharged
       , order_status.order_status as OrderStatus
FROM order_detail
INNER JOIN service_order on service_order.service_order_id =
order_detail.service_order_id
INNER JOIN ultimutt_walk_customer on ultimutt_walk_customer.ultimutt_walk_customer_id =
service_order.ultimutt_walk_customer_id
INNER JOIN staff_member ON staff_member.staff_member_id = service_order.staff_member_id
INNER JOIN pet ON pet.pet_id = order_detail.pet_id
INNER JOIN order_status ON order_status.order_status_id=service_order.order_status_id
WHERE order_status.order_status_id = 2
GO
--double check there aren't any open orders when calling code
SELECT * FROM ClosedOrders

-- nothing appears, great!
-- now I will create some orders from last week so that I can see the whole process from
order to invoice to staff payment
```

```
INSERT INTO service_order(ultimutt_walk_customer_id, order_status_id, staff_member_id,
order_date)
VALUES ((SELECT ultimutt_walk_customer_id FROM ultimutt_walk_customer WHERE
telephone_number= '9546085634'), 1, (SELECT staff_member_id FROM staff_member WHERE
first_name = 'Gary'), '8/27/2019')
       , ((SELECT ultimutt_walk_customer_id FROM ultimutt_walk_customer WHERE
telephone_number= '9546085634'), 1, (SELECT staff_member_id FROM staff_member WHERE
first_name = 'Katie'), '8/30/2019')

--find service_order_id number
SELECT * FROM
service_order
WHERE ultimutt_walk_customer_id =(SELECT ultimutt_walk_customer_id FROM
ultimutt_walk_customer WHERE telephone_number= '9546085634')

--insert into the details
INSERT INTO order_detail(service_order_id, requested_service_date, amount, pet_id)
VALUES (21, '8/27/2019', 15, (SELECT pet_id FROM pet WHERE pet_name ='Barlo'))
              , (22, '8/30/2019', 15, (SELECT pet_id FROM pet WHERE pet_name = 'Barlo'))

-- create procedure to close out an order
-- much easier

GO
CREATE PROCEDURE CloseOutOrder(@service_order_ID int)
AS
BEGIN
       UPDATE service_order SET order_status_id = 2
       WHERE service_order_id = @service_order_ID
END

GO
DECLARE @closedorder int
SET @closedorder = (SELECT service_order_id FROM service_order WHERE service_order_id=21)
EXEC CloseOutOrder @closedorder
-- check if it worked!
SELECT * FROM service_order WHERE service_order_id = 21
-- it did, close other order
DECLARE @secondclosedorder int
SET @secondclosedorder = (SELECT service_order_id FROM service_order WHERE
service_order_id=22)
EXEC CloseOutOrder @secondclosedorder

SELECT * FROM ClosedOrders
--so both of Barlo's walks are officially closed
```

	OrderNumber	OrderDate	ServiceDate	CustomerName	Pet	StaffMemberAssigned	AmountCharged	OrderStatus
1	21	2019-08-27 00:00:00.000	2019-08-27 00:00:00.000	Michael London	Barlo	Gary Guinta	15	CLOSED
2	22	2019-08-30 00:00:00.000	2019-08-30 00:00:00.000	Michael London	Barlo	Katie Hanks	15	CLOSED

```
--let us move on to payment

--insert INVOICE info
INSERT INTO invoice(service_order_id, invoice_status_id, datepaid, amount,
date_of_service)
VALUES (1,2,'9/8/2019',15,
'9/5/2019'),(2,1,'9/8/2019',15,'9/6/2019'),(10,1,'9/6/2019',6,'9/6/2019'),(11,2,'9/6/2019
```

```
', 6,'9/7/2019'),(12,2,'9/6/2019',15,'9/3/2019'),(13,2,'9/6/2019', 30,'9/4/2019'), (14,2,
'9/6/2019',15, '9/5/2019'),
       (15,2, '9/6/2019',15, '9/6/2019'), (16,2, '9/6/2019',6, '9/3/2019'), (17,2,
'9/6/2019',6, '9/4/2019'), (18,2, '9/6/2019',6,'9/5/2019'), (19,2, '9/6/2019',6,
'9/6/2019'), (20,2, '9/6/2019',6,'9/7/2019'), (21,2,'8/27/2019',15,'8/27/2019'),
(22,2,'8/30/2019',15, '8/30/2019')
--create view to see invoices / paid and not paid
GO
CREATE VIEW Invoices as
SELECT
       invoice.invoice_id as InvoiceNumber
       , service_order.service_order_id as ServiceNumber
       , order_detail.requested_service_date as DateofService
       , ultimutt_walk_customer.first_name + ' ' + ultimutt_walk_customer.last_name as
CustomerName
       , invoice.amount as AmountCharged
       , invoice_status.invoice_status as InvoiceStatus
FROM invoice
INNER JOIN service_order on service_order.service_order_id = invoice.service_order_id
INNER JOIN invoice_status on invoice_status.invoice_status_id = invoice.invoice_status_id
INNER JOIN ultimutt_walk_customer on ultimutt_walk_customer.ultimutt_walk_customer_id =
service_order.ultimutt_walk_customer_id
INNER JOIN order_detail on order_detail.service_order_id = service_order.service_order_id

--take a look at our view

GO
SELECT * FROM Invoices
```

	InvoiceNumber	ServiceNumber	DateofService	CustomerName	AmountCharged	InvoiceStatus
1	1	1	2019-09-05 00:00:00.000	Jose Ramos	15	PAID
2	2	2	2019-09-06 00:00:00.000	Jose Ramos	15	UNPAID
3	3	10	2019-09-06 00:00:00.000	Lily Kleinman	6	UNPAID
4	4	11	2019-09-07 00:00:00.000	Lily Kleinman	6	PAID
5	5	12	2019-09-03 00:00:00.000	Demi Lovato	15	PAID
6	6	13	2019-09-04 00:00:00.000	Demi Lovato	30	PAID
7	7	14	2019-09-05 00:00:00.000	Demi Lovato	15	PAID
8	8	15	2019-09-06 00:00:00.000	Demi Lovato	15	PAID
9	9	16	2019-09-03 00:00:00.000	Lily Kleinman	6	PAID
10	10	17	2019-09-04 00:00:00.000	Lily Kleinman	6	PAID
11	11	18	2019-09-05 00:00:00.000	Lily Kleinman	6	PAID

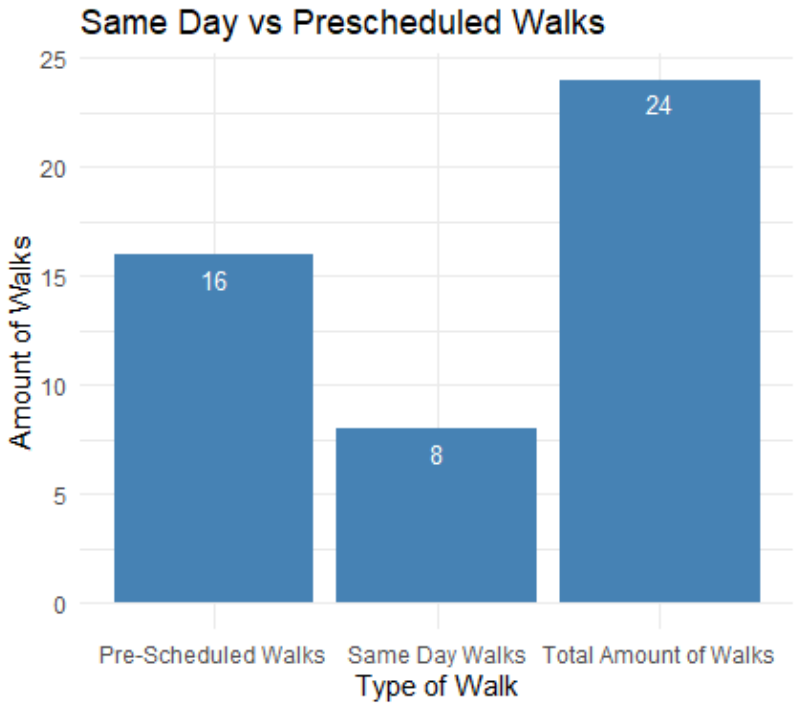
```
--now let's update our staff payments
INSERT INTO staff_payment(staff_member_id, amount, payment_date, service_order_id)
VALUES ((SELECT staff_member_id FROM staff_member WHERE first_name = 'GARY'),  15,
'8/31/2019', (SELECT service_order_id FROM service_order WHERE service_order_id = 2))

--now let's update our staff payments
INSERT INTO staff_payment(staff_member_id, amount, payment_date, service_order_id)
VALUES ((SELECT staff_member_id FROM staff_member WHERE first_name = 'GARY'),  15,
'8/31/2019', (SELECT service_order_id FROM service_order WHERE service_order_id = 2))

SELECT * FROM staff_payment

-- once again not useful so I am going to create a PROCEDURE for each employee to keep
track of their payments
```

```
GO
CREATE PROCEDURE ViewMyPayments(@employeeID int)
AS
BEGIN
	SELECT
	staff_member.first_name + ' ' + staff_member.last_name as EmployeeName
	,staff_payment.payment_date as DateofPayment
	, staff_payment.amount as AmountPaid
	, staff_member.first_name + ' ' + staff_member.last_name as EmployeeName
	, service_order.service_order_id as OrderID
	, order_detail.requested_service_date as DateofService
	, pet.pet_name as PetWalked
FROM staff_payment
INNER JOIN staff_member ON staff_member.staff_member_id = staff_payment.staff_member_id
INNER JOIN service_order ON service_order.service_order_id =
staff_payment.service_order_id
JOIN order_detail ON order_detail.service_order_id = service_order.service_order_id
JOIN pet ON pet.pet_id = order_detail.pet_id
END
--let's look at Gary Guinta's payments, staff ID = 2
GO
DECLARE @myID int
SET @myID = (SELECT staff_member_id FROM staff_member WHERE staff_member_id=2)
EXEC ViewMyPayments @myID

--view open orders
SELECT * FROM OpenOrders
-- need to close orders for gina due to hurricane, no invoice needed
--gina ID = 13
--close the rest of the orders out
SELECT * FROM service_order
WHERE ultimutt_walk_customer_id = 13
--service order 3-6 to be closed
DECLARE @closedorder int
SET @closedorder = (SELECT service_order_id FROM service_order WHERE service_order_id=3)
EXEC CloseOutOrder @closedorder
--next
DECLARE @secondclosedorder int
SET @secondclosedorder = (SELECT service_order_id FROM service_order WHERE
service_order_id=4)
EXEC CloseOutOrder @secondclosedorder
--next
DECLARE @thirdclosedorder int
SET @thirdclosedorder = (SELECT service_order_id FROM service_order WHERE
service_order_id=5)
EXEC CloseOutOrder @thirdclosedorder
-- next
DECLARE @fourthclosedorder int
SET @fourthclosedorder = (SELECT service_order_id FROM service_order WHERE
service_order_id=6)
EXEC CloseOutOrder @fourthclosedorder

--Gianni/Joel First, order 1,2
--close Linda's 7 8 9 10 16 17 18 19
--close Kristie's 12 13 14 15
GO
DECLARE @closedorder int
SET @closedorder = 1
```

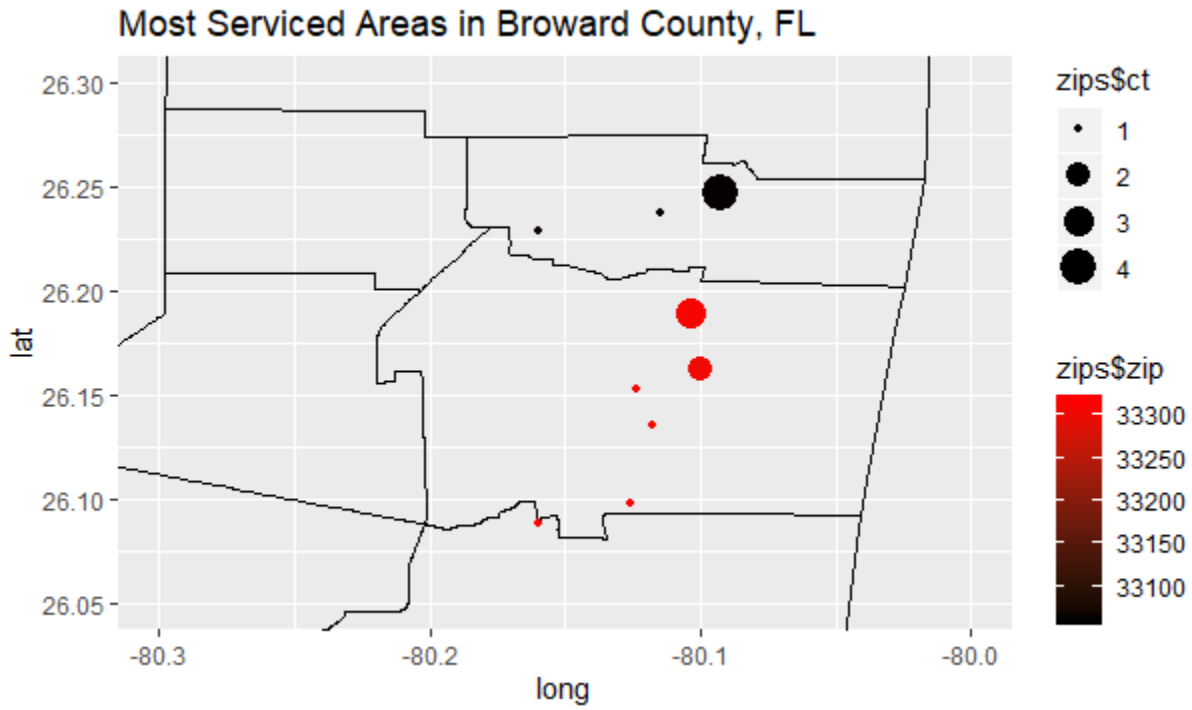
```
EXEC CloseOutOrder @closedorder
GO
DECLARE @closedorder int
SET @closedorder = 2
EXEC CloseOutOrder @closedorder

--Linda's
GO
DECLARE @closedorder int
SET @closedorder = 7
EXEC CloseOutOrder @closedorder
GO
DECLARE @closedorder int
SET @closedorder = 8
EXEC CloseOutOrder @closedorder
GO
DECLARE @closedorder int
SET @closedorder = 9
EXEC CloseOutOrder @closedorder
GO
DECLARE @closedorder int
SET @closedorder = 10
EXEC CloseOutOrder @closedorder
GO
DECLARE @closedorder int
SET @closedorder = 16
EXEC CloseOutOrder @closedorder
GO
DECLARE @closedorder int
SET @closedorder = 17
EXEC CloseOutOrder @closedorder
GO
DECLARE @closedorder int
SET @closedorder = 18
EXEC CloseOutOrder @closedorder
GO
DECLARE @closedorder int
SET @closedorder = 19
EXEC CloseOutOrder @closedorder

--Kristie's
GO
DECLARE @closedorder int
SET @closedorder = 12
EXEC CloseOutOrder @closedorder
GO
DECLARE @closedorder int
SET @closedorder = 13
EXEC CloseOutOrder @closedorder
GO
DECLARE @closedorder int
SET @closedorder = 14
EXEC CloseOutOrder @closedorder
GO
DECLARE @closedorder int
SET @closedorder = 15
EXEC CloseOutOrder @closedorder
--close out some more orders
```

```
GO
DECLARE @closedorder int
SET @closedorder = 11
EXEC CloseOutOrder @closedorder
GO
DECLARE @closedorder int
SET @closedorder = 15
EXEC CloseOutOrder @closedorder
GO
DECLARE @closedorder int
SET @closedorder = 20
EXEC CloseOutOrder @closedorder

--double check they closed out
SELECT * FROM ClosedOrders

--forgot an order of last week - Ted Rochester walks for his two dogs. Same Day Request
INSERT INTO service_order(ultimutt_walk_customer_id, order_status_id, staff_member_id,
order_date)
VALUES ((SELECT ultimutt_walk_customer_id FROM ultimutt_walk_customer WHERE
telephone_number= '9547893293'), 1, (SELECT staff_member_id FROM staff_member WHERE
first_name = 'Katie'), '9/5/2019')
       , ((SELECT ultimutt_walk_customer_id FROM ultimutt_walk_customer WHERE
telephone_number= '9547893293'), 1, (SELECT staff_member_id FROM staff_member WHERE
first_name = 'Katie'), '9/5/2019')

INSERT INTO dog_walk(start_time, end_time, miles_walked)
       VALUES ('9/5/2019 8:00PM', '9/5/2019 9:00PM', 1), ('9/5/2019 8:00PM', '9/5/2019
9:00PM', 1)
SELECT * FROM dog_walk

INSERT INTO order_detail(requested_service_date, amount, dog_walk_id, pet_id,
service_order_id)
       VALUES ('9/5/2019', 10, 1, 23, 23), ('9/5/2019', 10,2 , 24, 24)

INSERT INTO invoice(amount,date_of_service,datepaid,invoice_status_id,service_order_id)
       VALUES (10, '9/5/2019', '9/5/2019', 2, 23),(10, '9/5/2019', '9/5/2019', 2, 24)

--Data Question 1: How many walks are prescheduled vs. scheduled on same day
SELECT
       (SELECT
       COUNT(requested_service_date)
FROM order_detail OD
JOIN service_order SO
ON SO.service_order_id = OD.service_order_id
WHERE OD.requested_service_date = SO.order_date) as SameDayWalks,
(
SELECT
       COUNT(requested_service_date)
FROM order_detail OD
JOIN service_order SO
ON SO.service_order_id = OD.service_order_id
WHERE OD.requested_service_date != SO.order_date) as PreScheduledWalks,
(
SELECT
       COUNT(requested_service_date)) as TotalWalks
FROM order_detail
--total out of 24
```

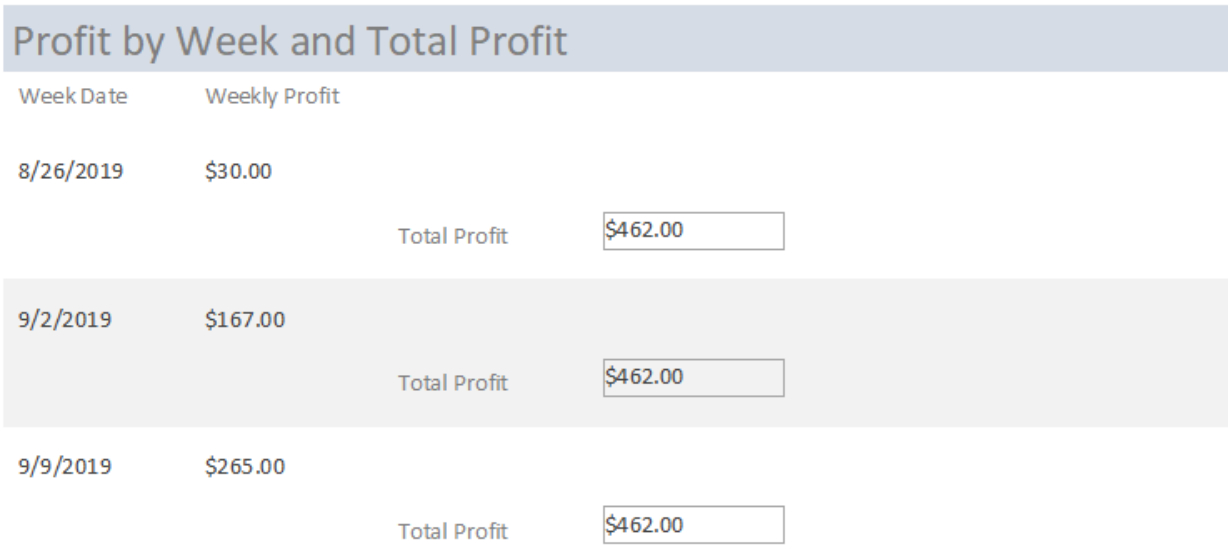
```
--8/24 or 1/3 are scheduled same-day
--16/24 or 2/3 are pre-scheduled
--create Bar Chart in R
```



```sql
--Data Question 2: What are the most serviced zip codes?
--create choropleth map
--create Query for R to select zip codes

SELECT
       home_zip
FROM ultimutt_walk_customer
```



```sql
--Data Question 3: What are the profits of the company at any given moment?


--weekly profit function, business week starts on Monday
--create VIEW

GO

CREATE VIEW weekly_profit AS
SELECT
       DATEADD(week, DATEDIFF(week, 0, date_of_service), 0) 'Week_Begin'
       ,SUM(amount) as weekly_profit
  FROM
  invoice
  GROUP BY
  DATEADD(week, DATEDIFF(week, 0, date_of_service), 0)


GO
SELECT * FROM weekly_profit

GO
CREATE VIEW totalprofit AS
SELECT
       SUM(amount) as TotalProfit,
       GETDATE() as ToDate
FROM invoice
WHERE invoice_status_id=2
GO
SELECT * FROM totalprofit

--create report from Access
```



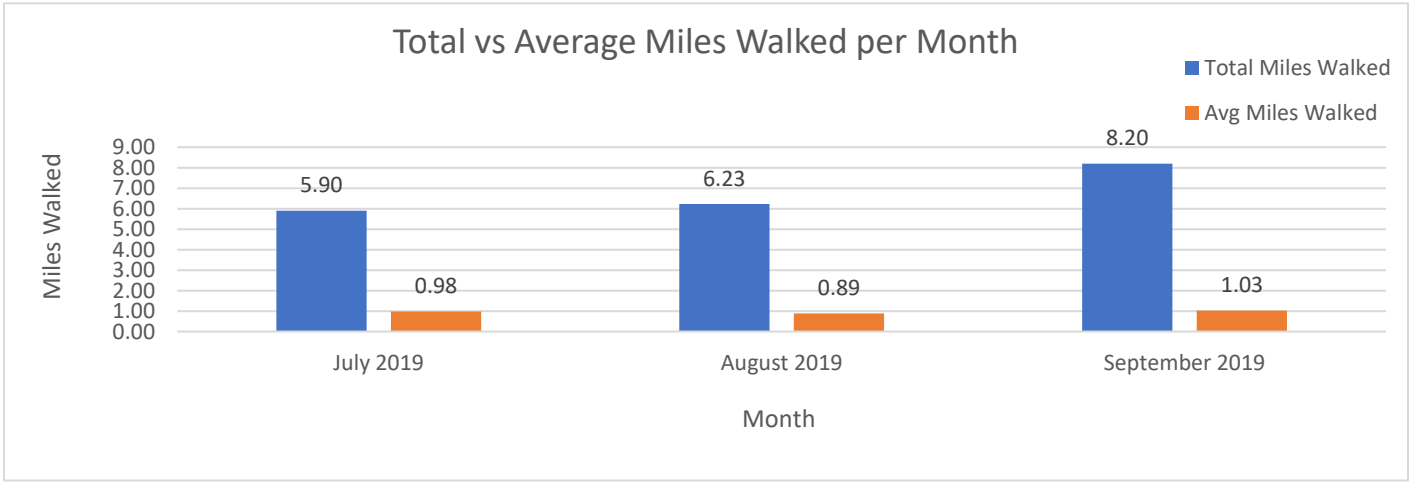
```
--Data Question 4:   What is the average length of a dog walk? How many miles are walked
a month?

--use Form to Enter Dog Walk Info
```

Dog Walk Form

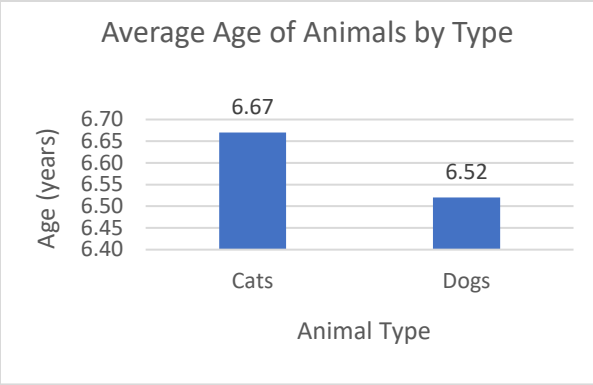
Dog Walk ID	Start Time	End Time	Miles Walked
1	9/5/2019	9/5/2019	1
2	9/5/2019	9/5/2019	1
20	7/27/2019	7/20/2019	3.5
21	7/28/2019	7/28/2019	0.35
{New}			

```
--info for dog_walk
SELECT
requested_service_date,
pet.pet_name
FROM order_detail
INNER JOIN pet ON pet.pet_id = order_detail.pet_id

--what is average length of dog walks?
SELECT
       AVG(miles_walked)
FROM dog_walk
--average length is .97 miles
-- report of miles walked per month
```

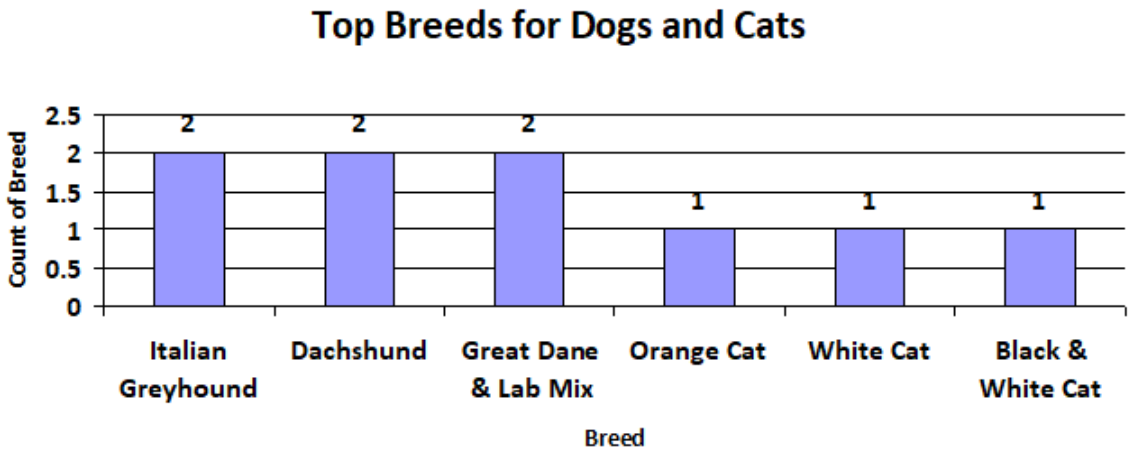
Total and Average Miles Walked Per Month		
Month	**Total Miles Walked**	**Avg Miles Walked**
August 2019	5.9	0.98
July 2019	6.23	0.89
September 2019	8.2	1.03



```sql
--Data Question 5: What is the average age of the pets and most common breed?
SELECT
       animal_type,
       AVG(age) as AverageAgeOfPets
FROM pet
GROUP BY animal_type
```




```sql
SELECT
TOP 3 pet_breed
,COUNT(pet_breed) as Count_of_Breed
FROM pet
WHERE animal_type = 'dog'
GROUP BY pet_breed
ORDER BY COUNT(*) DESC

SELECT
TOP 3 pet_breed
,COUNT(pet_breed) as Count_of_Breed
FROM pet
WHERE animal_type = 'cat'
GROUP BY pet_breed
ORDER BY COUNT(*) DESC

---TOP DOG breeds are Italian Greyhounds, Dachshunds, and Great Dane & Lab Mix
--- Only 3 cat breeds so far - Orange, White, White Black
```



Reflection

My first and only assumption in this class was that the theory portion, including the conceptual and logical model, wasn't useful. However, upon reflection I understand the importance of the conceptual and especially the logical model. I don't think I would have been able to complete my project in a timely manner if I didn't have the model to map everything out. If there is anything I could have done differently that would have been collected data more meticulously and for a longer period of time, I felt towards the end I ran out of time so I wasn't able to insert all of the data I collected. If I had collected more data I believe my project would have been more insightful at identifying trends and more detailed profit information. Also, I would have separated the dates into separate columns as well to track trends more in depth, it was hard to do it by the 'mm/dd/yyyy' format. I believe this is a good lesson for me as a data science that the more data the better. Also, learning how to connect a database to R and Microsoft Access was extremely helpful and I will be able to use that in the future.

Summary

Developing the business rules and then the logical model were extremely crucial in the implementation of the database to create tables, the data types, and relationships between them. Also, the logical model was helpful for mapping the relationships in Microsoft Access. Most of my data questions were answered by a combination of SQL SELECT statements and creating a chart in either Access or R. Microsoft Access was the easiest to use for generating forms and made data entry very easy so that the answers to my data questions could be as insightful as possible. But, I believe R was much easier to use for generating plots and the choropleth map. I also believe the reports generated by R were much more aesthetically pleasing. It would have been much more helpful to have learned the connection to Access and R earlier on in the course and in a little more detail to utilize the tools to the ultimate potential.