

**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.

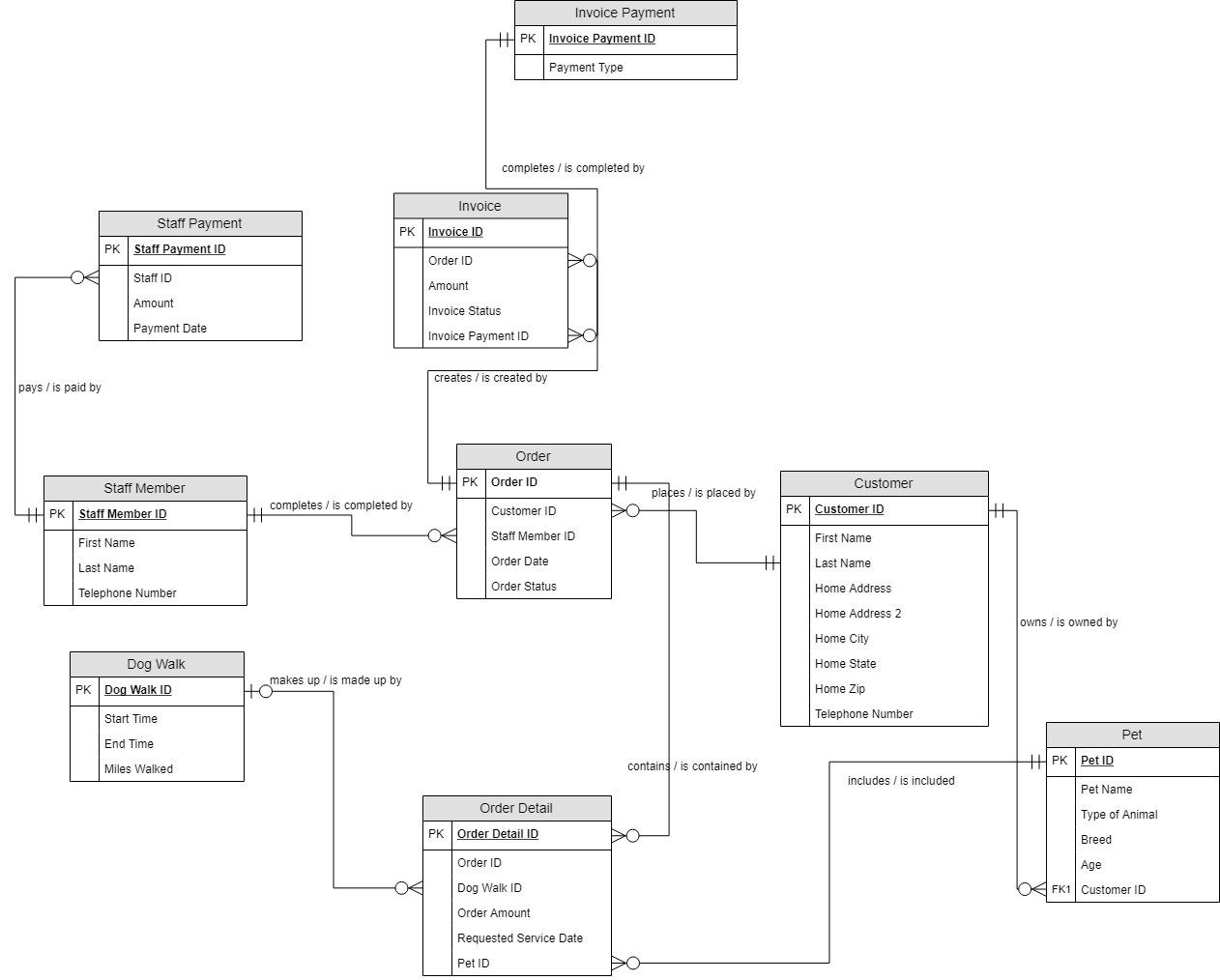
A **PAID invoice status** is when Ultimutt Walks has received payment for a completed order.

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

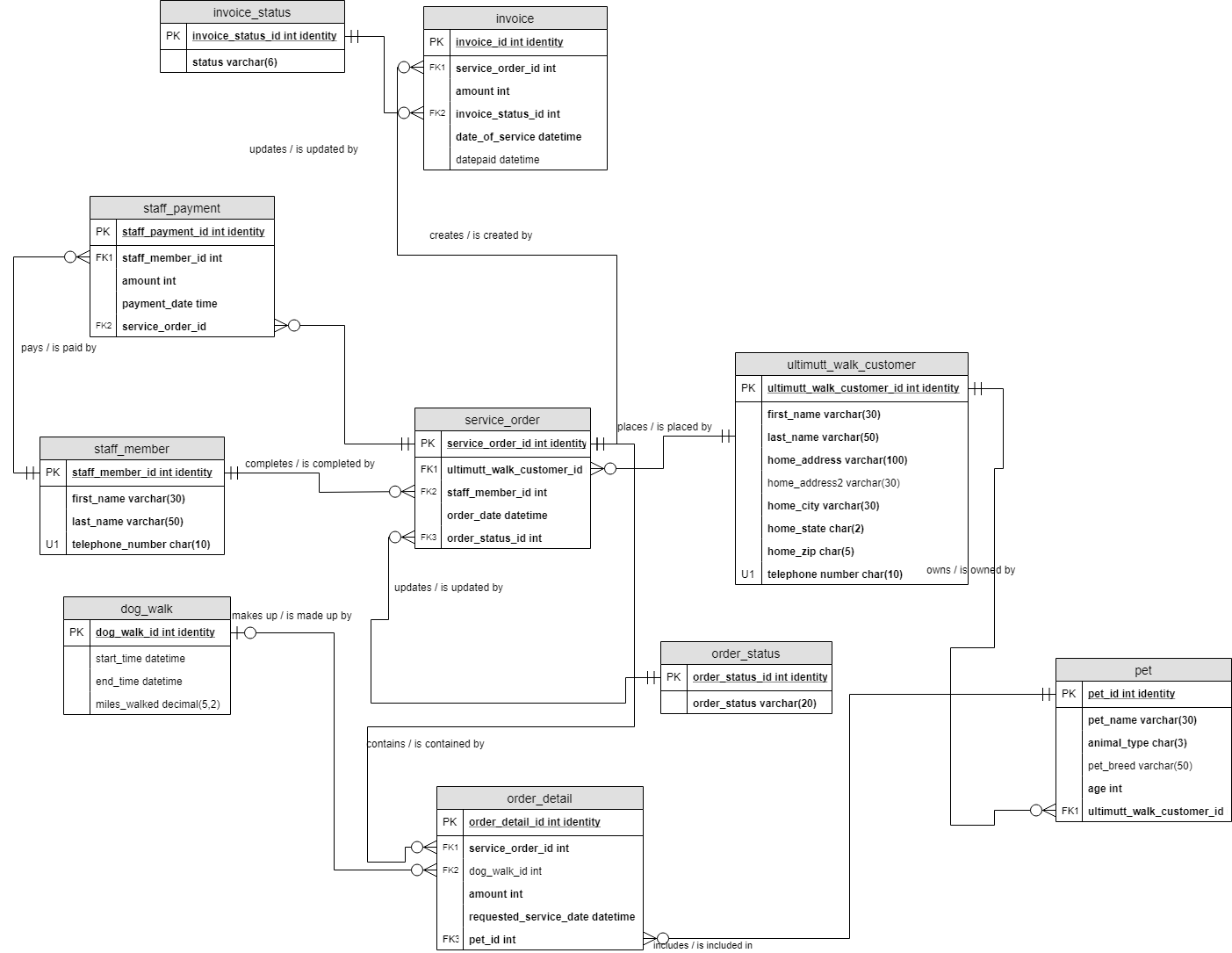
**Data Questions**

* 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**

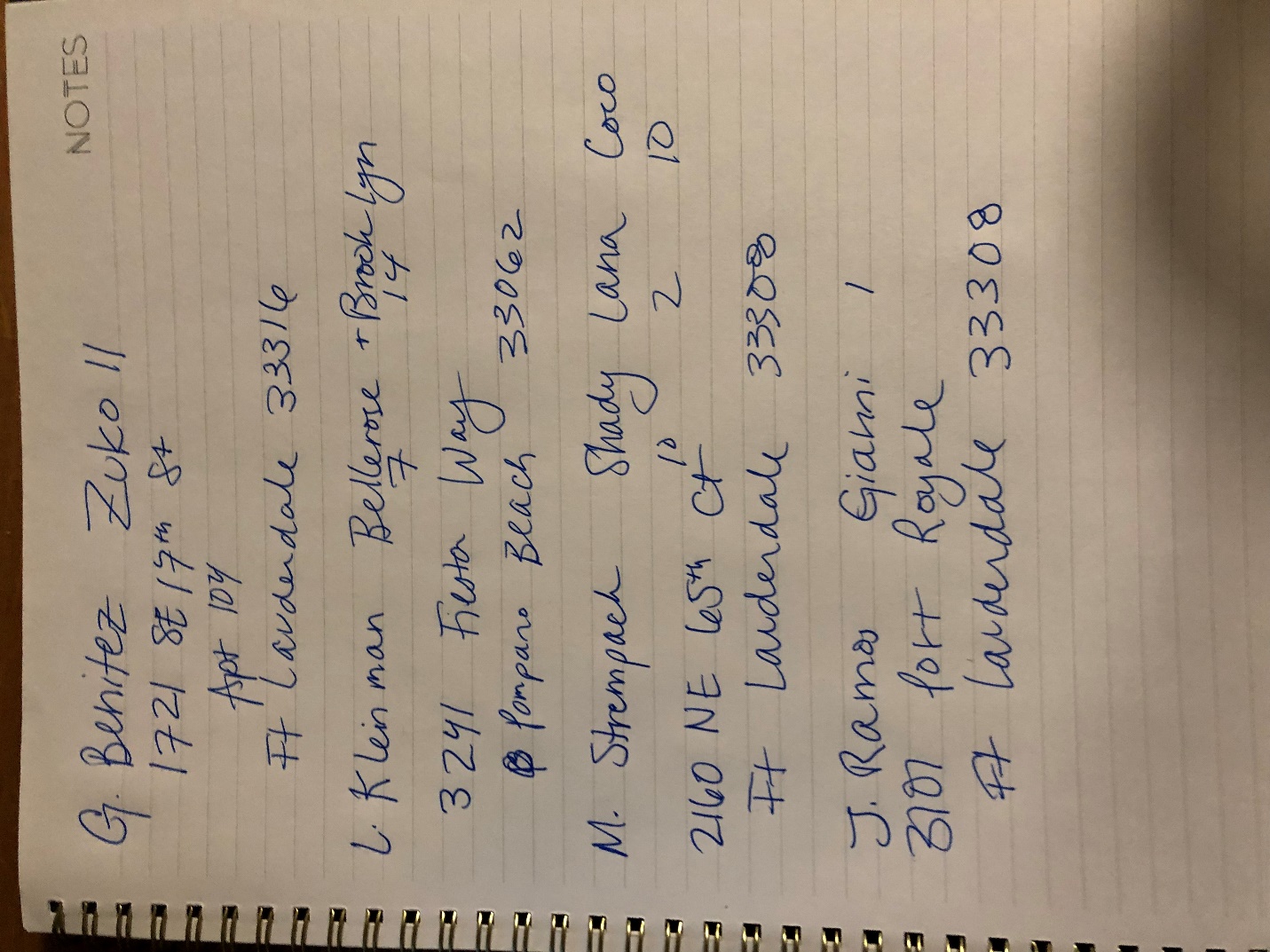
****

**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



--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,

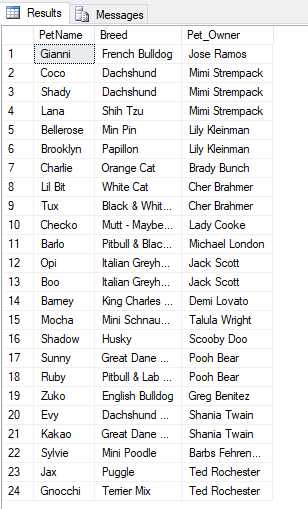
ultimutt\_walk\_customer.first\_name + ' ' + ultimutt\_walk\_customer.last\_name as Pet\_Owner

FROM pet

INNER JOIN ultimutt\_walk\_customer ON pet.ultimutt\_walk\_customer\_id = ultimutt\_walk\_customer.ultimutt\_walk\_customer\_id

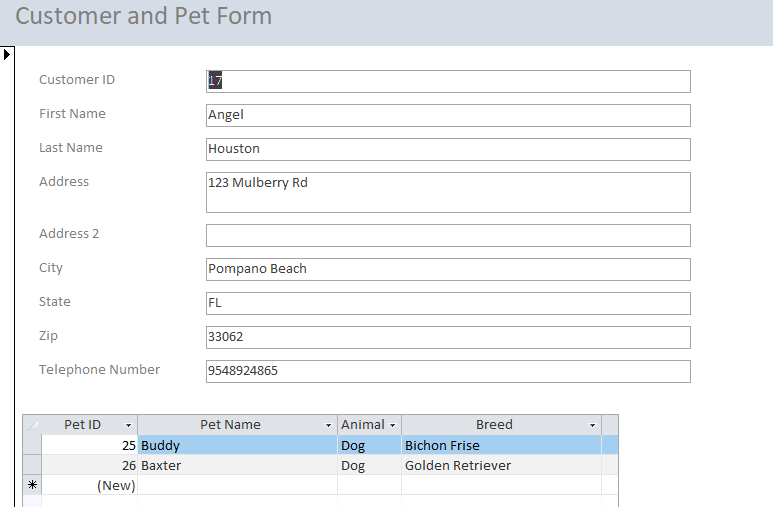
GO

SELECT\*FROM PetandOwners



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 ('Arn', 'Lodaa', '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

--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 = 'Arn'), '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 = 'Arn'), '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 = 'Arn'), '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 = 'Arn'), '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 = 'Arn'), '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 = 'Arn'), '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 = 'Arn'), '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 = 'Arn'), '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 = 'Arn'), '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 = 'Arn'), '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 = 'Arn'), '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 = 'Arn'), '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 = 'Arn'), '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 = 'Arn'), '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 = 'Arn'), '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 = 'Arn'), '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 = 'Arn'), '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 = 'Arn'), '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 = 'Arn'), '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 = 'Arn'), '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

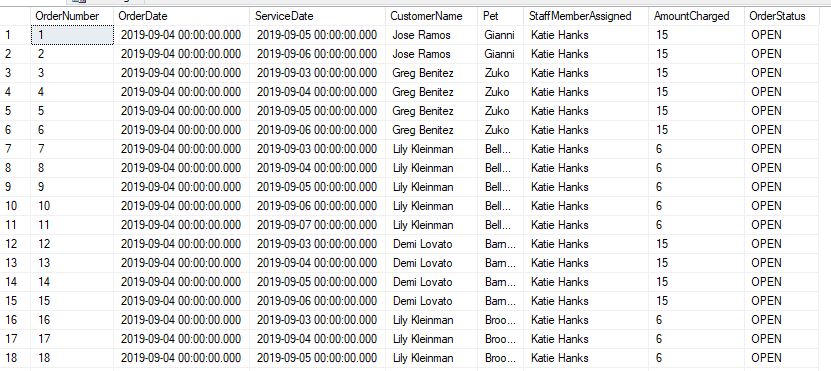
INNER JOIN order\_status ON order\_status.order\_status\_id=service\_order.order\_status\_id

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



--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 = 'Arn'), '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

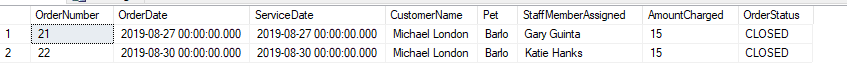
DECLARE @secondclosedorder int

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



--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

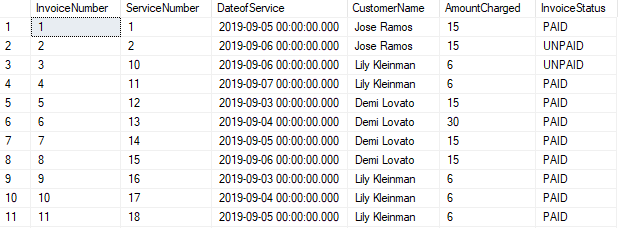
INNER JOIN ultimutt\_walk\_customer on ultimutt\_walk\_customer.ultimutt\_walk\_customer\_id = service\_order.ultimutt\_walk\_customer\_id

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



--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 = 'Arn'), '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 = 'Arn'), '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

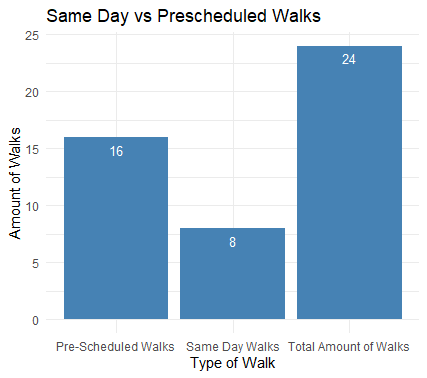
FROM order\_detail

--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



--Data Question 2: What are the most serviced zip codes?

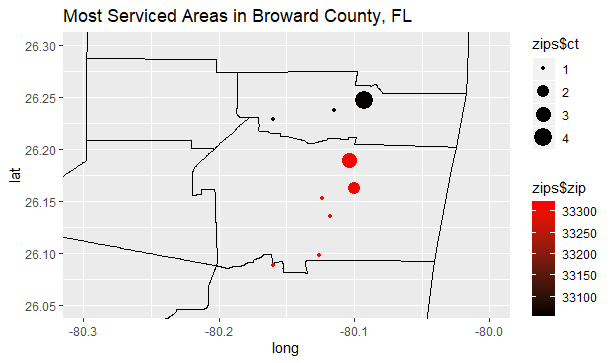
--create choropleth map

--create Query for R to select zip codes

SELECT

home\_zip

FROM ultimutt\_walk\_customer



--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

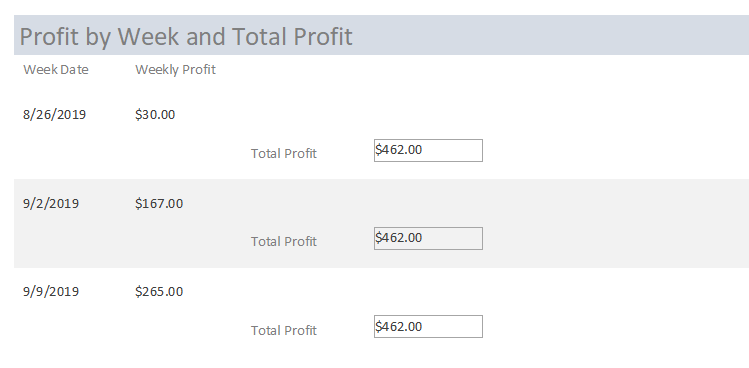
FROM invoice

WHERE invoice\_status\_id=2

GO

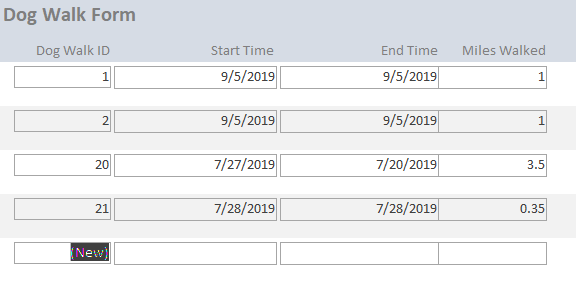
SELECT \* FROM totalprofit

--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



--info for dog\_walk

SELECT

requested\_service\_date,

| **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 |

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

--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

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

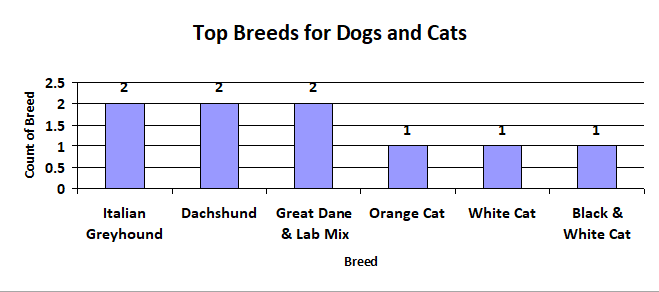
FROM pet

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.