Richard Sangster
IST659 M407 Project
Spring 2022

J&J Autoworks

Business Database

Table of Contents

Part One 3
- Summary 3
- Stakeholders 3
- Business Rules 3
- Data Questions 3
- Conceptual Model 3
- Logical Model 4

Part 2 5
- Data Definition Language – Creating Tables and Constraints 5
- Data Manipulation Language 5
  - Adding Data using INSERT Statements 6
  - Querying Data Using SELECT Statements 9
- Programming Objects 10
- User Interface 11
- Reflection 13

Richard Sangster
IST659 M407 Project
Spring 2022

Part One

Summary

J&J Autoworks has more than 25 years of experience in foreign car sales and service. With a small shop like this, tracking and processing of services has never been defined. J&J would like to develop a database to record and analyze service data. In doing so they will be able to analyze their data to perform more accurate examinations and estimates of time and cost to customers.

Note: J&J exists and operates as expressed, however all data in this document is generated for academic use for purposes of displaying database functionality and effectiveness.

Stakeholders

- New Customers
    - People who need to get service done on their vehicles
- J&J Employees and Owner
    - Mechanics who are budgeting time, Owner performing capacity and cost analyses

Business Rules

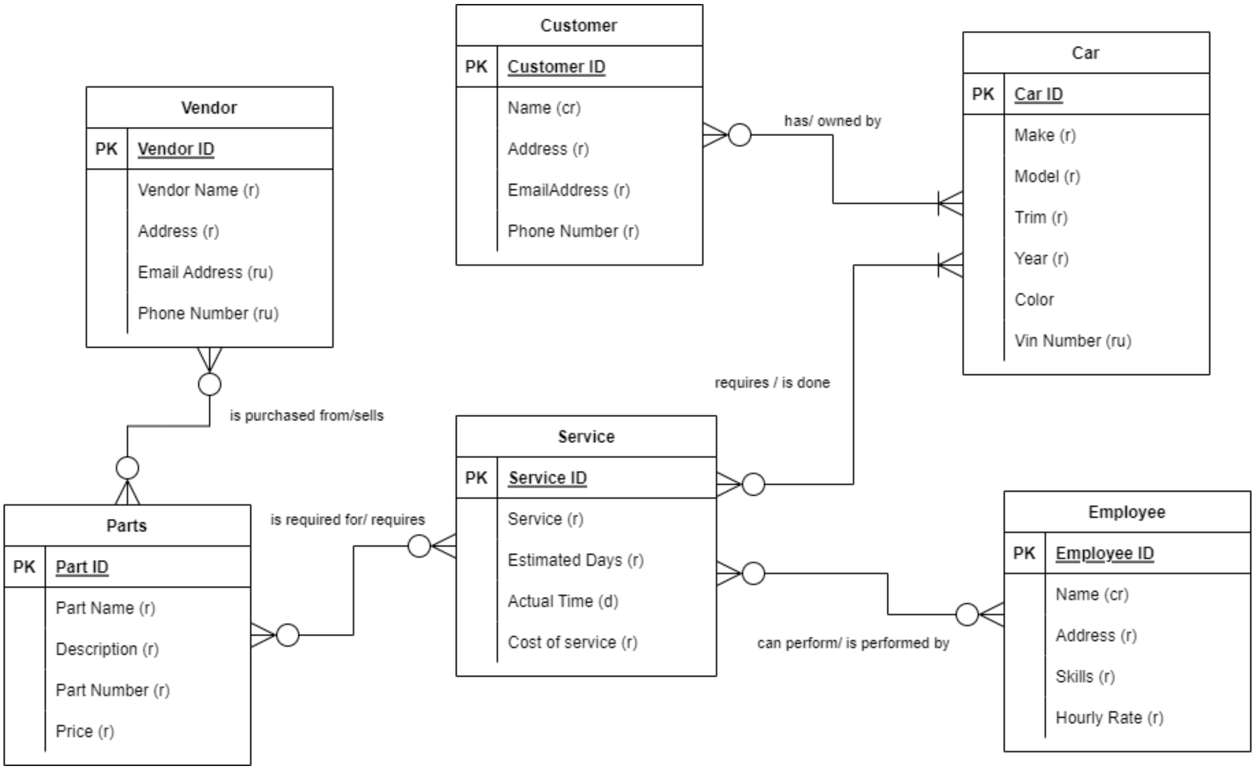
- Each customer must record their address, email, and phone number to allow for proper contact.
- Each customers car is inspected upon being dropped off and an expected time of completion is given to the customer. A quote for the cost of the service is also given to allow the customer to decide whether or not they want the service performed.
- Each customer can have many cars at the shop and each car can be owned by multiple customers (i.e. wife and husband).
- Each employee has been trained with different skills and abilities. Each employee also has their own pay rate for their time and experience. An employee can perform many services on a car and different services can be performed by different employees.
- Each car is recorded by its make, model, trim, year and color. The cars vin number is also recorded and unique to that car.
- Once the service is completed the amount of time and total cost of service is recorded for future budgeting.
- As part of the business, every car may have an owner but every owner must have a car.
- Other business rules will be defined in later iterations of the database.

Data Questions

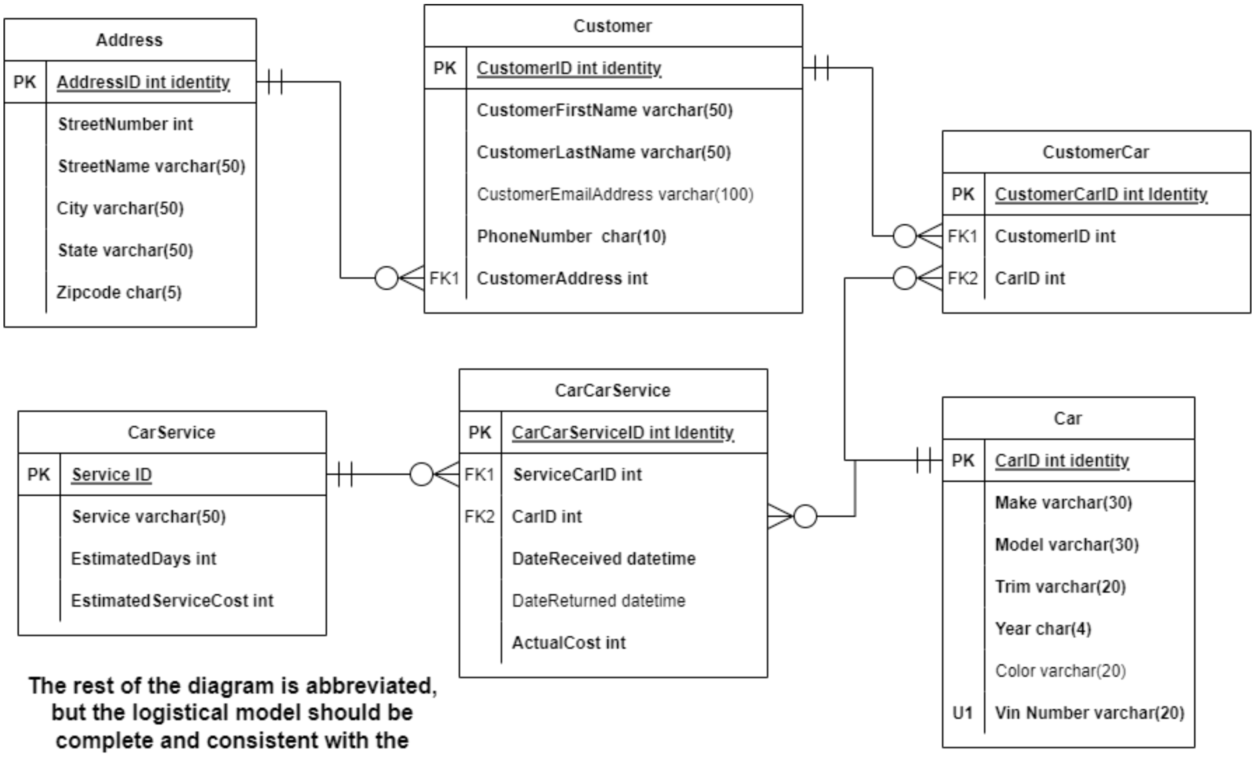
- Who are the customers and how can we contact them?
- What kinds of service did we do on each vehicle? How long does each service take?
- What kinds of services are performed on certain vehicles?
- How many times has a certain employee performed the same service? How often do they work on a certain vehicle?

Richard Sangster
IST659 M407 Project
Spring 2022

Conceptual Model



Logical Model



The rest of the diagram is abbreviated, but the logistical model should be complete and consistent with the conceptual model and tables in Part 2

Part 2

Data Definition Language – Creating Tables and Constraints

```
/*
Author : Richard Sangster
Course : IST659 M407
Term   : Spring 2022
Name   : Project J&J
*/

--Begin by dropping all tables from database
DROP TABLE IF EXISTS CustomerCar
DROP TABLE IF EXISTS CarCarService
DROP TABLE IF EXISTS Customer
DROP TABLE IF EXISTS CustomerAddress
DROP TABLE IF EXISTS CarService
DROP TABLE IF EXISTS Car

-- Creating Tables
Create Table CustomerAddress (
-- Placing in columns
       CustomerAddressID int identity primary key,
       StreetNumber int not null,
       StreetName varchar(50) not null,
       City varchar(50) not null,
       AddressState varchar(50) not null, --Name changed due to State being a system term
       Zipcode char(5) not null
)

Create Table CarService (
       CarServiceID int identity primary key,
       ServiceDone varchar(50) not null,
       EstimatedDays int not null,
       EstimatedServiceCost int not null
)

Create Table Car (
       CarID int identity primary key,
       Make varchar(30) not null,
       Model varchar(30) not null,
       CarTrim varchar(20) not null,
       CarYear char(4) not null,
       Color varchar(20),
       VinNumber varchar(20) not null Unique
)

Create Table Customer (
       CustomerID int identity primary key,
       CustomerFirstName varchar(50) not null,
       CustomerLastName varchar(50) not null,
       CustomerEmailAddress varchar(100),
       PhoneNumber char(10) not null,
       CustomerAddressID int not null,
       Constraint F1_Customer Foreign Key (CustomerAddressID) References
CustomerAddress(CustomerAddressID)
)
```

```
Create Table CarCarService (
        ServiceCarID int identity primary key,
        ServiceID int not null Foreign Key References CarService(CarServiceID),
        CarID int not null Foreign Key References Car(CarID),
        DateReceived datetime not null,
        DateReturned datetime,
        ActualCost int
)

Create Table CustomerCar (
        CustomerCarID int identity primary key,
        CustomerID int not null,
        CarID int not null,
        Constraint F1_CustomerCar Foreign Key (CustomerID) References
Customer(CustomerID),
        Constraint F2_CustomerCar Foreign Key (CarID) References Car(CarID)
)
```

Data Manipulation Language

Adding Data using INSERT Statements

```
--Inserting into tables, abbreviated to save space --
Insert Into CustomerAddress (StreetNumber, StreetName, City, AddressState, Zipcode)
values ('58', 'Tennyson', 'Detroit', 'Michigan', '48267');
Insert Into CustomerAddress (StreetNumber, StreetName, City, AddressState, Zipcode)
values ('860', 'Cambridge', 'Galveston', 'Texas', '77554');
Insert Into CustomerAddress (StreetNumber, StreetName, City, AddressState, Zipcode)
values ('211', 'Del Sol', 'Iowa City', 'Iowa', '52245');
Insert Into CustomerAddress (StreetNumber, StreetName, City, AddressState, Zipcode)
values ('17', 'Reindahl', 'New Haven', 'Connecticut', '06533');
Insert Into CustomerAddress (StreetNumber, StreetName, City, AddressState, Zipcode)
values ('9', 'Florence', 'Philadelphia', 'Pennsylvania', '19151');
Insert Into CustomerAddress (StreetNumber, StreetName, City, AddressState, Zipcode)
values ('456', 'Bartelt', 'Milwaukee', 'Wisconsin', '53277');
Insert Into CustomerAddress (StreetNumber, StreetName, City, AddressState, Zipcode)
values ('39', 'Forest Run', 'Little Rock', 'Arkansas', '72215');
Insert Into CustomerAddress (StreetNumber, StreetName, City, AddressState, Zipcode)
values ('6', 'Sage', 'Amarillo', 'Texas', '79159');
Insert Into CustomerAddress (StreetNumber, StreetName, City, AddressState, Zipcode)
values ('9', 'Westridge', 'Waterbury', 'Connecticut', '06721');
Insert Into CustomerAddress (StreetNumber, StreetName, City, AddressState, Zipcode)
values ('74', 'Fordem', 'Ogden', 'Utah', '84403');
--etc.--

--Abbreviated to save space --
Insert Into Customer (CustomerFirstName, CustomerLastName, CustomerEmailAddress,
PhoneNumber, CustomerAddressID) values ('Lorita', 'Tuddenham',
'ltuddenham0@washington.edu', '9683710281', 31);
Insert Into Customer (CustomerFirstName, CustomerLastName, CustomerEmailAddress,
PhoneNumber, CustomerAddressID) values ('Frederich', 'Elgood', 'felgood1@jimdo.com',
'1753172547', 2);
Insert Into Customer (CustomerFirstName, CustomerLastName, CustomerEmailAddress,
PhoneNumber, CustomerAddressID) values ('Mata', 'Iwanczyk', 'miwanczyk2@wsj.com',
'6944503384', 3);
```

```
Insert Into Customer (CustomerFirstName, CustomerLastName, CustomerEmailAddress,
PhoneNumber, CustomerAddressID) values ('Melloney', 'Whitehair',
'mwhitehair3@opensource.org', '6646238251', 23);
Insert Into Customer (CustomerFirstName, CustomerLastName, CustomerEmailAddress,
PhoneNumber, CustomerAddressID) values ('Hakim', 'Bottom', 'hbottom4@nytimes.com',
'8169651334', 44);
Insert Into Customer (CustomerFirstName, CustomerLastName, CustomerEmailAddress,
PhoneNumber, CustomerAddressID) values ('Lennie', 'Father', 'lfather5@smugmug.com',
'2107370996', 4);
Insert Into Customer (CustomerFirstName, CustomerLastName, CustomerEmailAddress,
PhoneNumber, CustomerAddressID) values ('Cati', 'Beatey', 'cbeatey6@virginia.edu',
'2269382816', 33);
Insert Into Customer (CustomerFirstName, CustomerLastName, CustomerEmailAddress,
PhoneNumber, CustomerAddressID) values ('Karita', 'Semiras',
'ksemiras7@biblegateway.com', '5009857037', 11);
Insert Into Customer (CustomerFirstName, CustomerLastName, CustomerEmailAddress,
PhoneNumber, CustomerAddressID) values ('Berthe', 'Easby', 'beasby8@seesaa.net',
'7463532177', 43);
Insert Into Customer (CustomerFirstName, CustomerLastName, CustomerEmailAddress,
PhoneNumber, CustomerAddressID) values ('Vick', 'Menlove', 'vmenlove9@google.pl',
'5913930493', 24);
--etc.--


--Abbreviated to save space --
Insert Into Car (Make, Model, CarTrim, CarYear, Color, VinNumber) values ('Volvo', 'S40',
'GR', 2001, 'Mauv', 'JHMZE2H59AS022197');
Insert Into Car (Make, Model, CarTrim, CarYear, Color, VinNumber) values ('Plymouth',
'Horizon', 'RU', 1978, null, 'JHMZF1C41BS266286');
Insert Into Car (Make, Model, CarTrim, CarYear, Color, VinNumber) values ('Ford', 'Flex',
'PE', 2010, 'Red', '1C3BC7EG4BN912462');
Insert Into Car (Make, Model, CarTrim, CarYear, Color, VinNumber) values ('Oldsmobile',
'Silhouette', 'AF', 1999, 'Teal', 'JN8AF5MR6ET803317');
Insert Into Car (Make, Model, CarTrim, CarYear, Color, VinNumber) values ('Land Rover',
'Freelander', 'DO', 2010, 'Pink', '1FTEW1C86AK106466');
Insert Into Car (Make, Model, CarTrim, CarYear, Color, VinNumber) values ('Lexus', 'GX',
'CF', 2006, 'Pink', 'JTDKN3DU9B0983871');
Insert Into Car (Make, Model, CarTrim, CarYear, Color, VinNumber) values ('Porsche',
'Cayenne', 'RU', 2004, 'Maroon', 'JH4CL96917C039167');
Insert Into Car (Make, Model, CarTrim, CarYear, Color, VinNumber) values ('Honda',
'Prelude', 'ID', 2001, 'Orange', 'WDDDJ7CB2BA322579');
Insert Into Car (Make, Model, CarTrim, CarYear, Color, VinNumber) values ('Ford', 'LTD
Crown Victoria', 'FI', 1988, 'Maroon', 'ML32A3HJ4FH601896');
Insert Into Car (Make, Model, CarTrim, CarYear, Color, VinNumber) values ('Buick',
'LaCrosse', 'FR', 2007, 'Red', 'WAUHF98P27A375784');

--etc.--



Insert Into CarService (ServiceDone, EstimatedDays, EstimatedServiceCost) values ('Oil
Change', 1, 100);
Insert Into CarService (ServiceDone, EstimatedDays, EstimatedServiceCost) values
('Transmission Flush', 2, 200);
Insert Into CarService (ServiceDone, EstimatedDays, EstimatedServiceCost) values ('Engine
Rebuild', 15, 7500);
Insert Into CarService (ServiceDone, EstimatedDays, EstimatedServiceCost) values ('Tires
Mounted and Balanced', 2, 50);
```

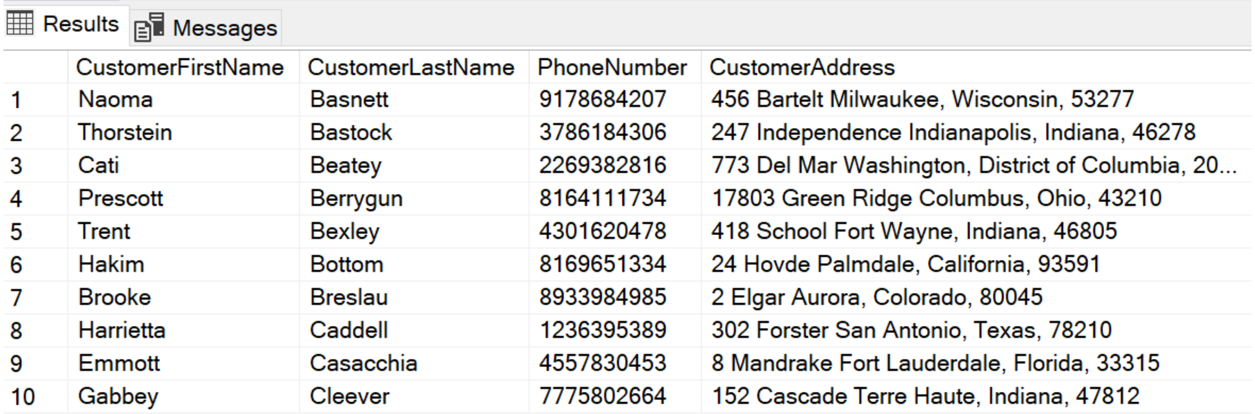
```sql
Insert Into CarService (ServiceDone, EstimatedDays, EstimatedServiceCost) values
('Replace Brakes', 1, 75);
Insert Into CarService (ServiceDone, EstimatedDays, EstimatedServiceCost) values
('Suspension Replacement', 7, 3000);

--Abbreviated to save space --
Insert Into CarCarService (ServiceID, CarID, DateReceived, DateReturned, ActualCost)
values (1, 1, '07/20/2021', '07/22/2021', 104);
Insert Into CarCarService (ServiceID, CarID, DateReceived, DateReturned, ActualCost)
values (1, 29, '12/29/2021', '12/30/2021', 83);
Insert Into CarCarService (ServiceID, CarID, DateReceived, DateReturned, ActualCost)
values (1, 3, '03/15/2022', '03/17/2022', 110);
Insert Into CarCarService (ServiceID, CarID, DateReceived, DateReturned, ActualCost)
values (1, 15, '02/23/2022', '02/23/2022', 87);
Insert Into CarCarService (ServiceID, CarID, DateReceived, DateReturned, ActualCost)
values (1, 8, '11/15/2021', '11/16/2021', 83);
Insert Into CarCarService (ServiceID, CarID, DateReceived, DateReturned, ActualCost)
values (1, 9, '03/31/2022', '04/01/2022', 95);
Insert Into CarCarService (ServiceID, CarID, DateReceived, DateReturned, ActualCost)
values (1, 13, '12/20/2021', '12/21/2021', 107);
Insert Into CarCarService (ServiceID, CarID, DateReceived, DateReturned, ActualCost)
values (1, 15, '11/16/2021', '11/16/2021', 87);
Insert Into CarCarService (ServiceID, CarID, DateReceived, DateReturned, ActualCost)
values (1, 5, '08/07/2021', '08/09/2021', 80);
Insert Into CarCarService (ServiceID, CarID, DateReceived, DateReturned, ActualCost)
values (1, 18, '01/02/2022', '01/05/2022', 119);

--etc.--


--Abbreviated to save space --
Insert Into CustomerCar (CustomerID, CarID) values (1, 15);
Insert Into CustomerCar (CustomerID, CarID) values (2, 17);
Insert Into CustomerCar (CustomerID, CarID) values (3, 11);
Insert Into CustomerCar (CustomerID, CarID) values (4, 12);
Insert Into CustomerCar (CustomerID, CarID) values (5, 16);
Insert Into CustomerCar (CustomerID, CarID) values (6, 19);
Insert Into CustomerCar (CustomerID, CarID) values (7, 10);
Insert Into CustomerCar (CustomerID, CarID) values (8, 18);
Insert Into CustomerCar (CustomerID, CarID) values (9, 14);
Insert Into CustomerCar (CustomerID, CarID) values (10, 13);

--etc.–
```

Richard Sangster
IST659 M407 Project
Spring 2022

Querying Data Using SELECT Statements

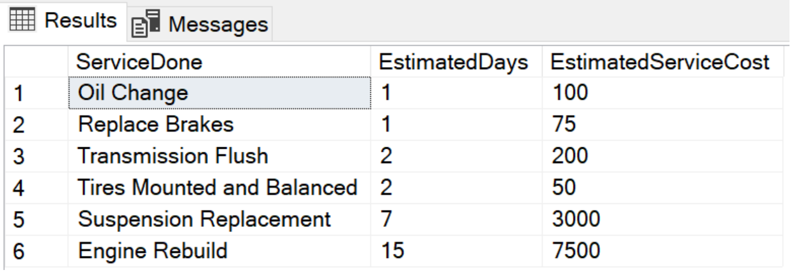
Note: We're also satisfying the Programming Objects requirements by creating views at the same time

```
SELECT * FROM CustomerInfo
ORDER BY CustomerLastName
GO
-- Results:
```



	CustomerFirstName	CustomerLastName	PhoneNumber	CustomerAddress
1	Naoma	Basnett	9178684207	456 Bartelt Milwaukee, Wisconsin, 53277
2	Thorstein	Bastock	3786184306	247 Independence Indianapolis, Indiana, 46278
3	Cati	Beatey	2269382816	773 Del Mar Washington, District of Columbia, 20...
4	Prescott	Berrygun	8164111734	17803 Green Ridge Columbus, Ohio, 43210
5	Trent	Bexley	4301620478	418 School Fort Wayne, Indiana, 46805
6	Hakim	Bottom	8169651334	24 Hovde Palmdale, California, 93591
7	Brooke	Breslau	8933984985	2 Elgar Aurora, Colorado, 80045
8	Harrietta	Caddell	1236395389	302 Forster San Antonio, Texas, 78210
9	Emmott	Casacchia	4557830453	8 Mandrake Fort Lauderdale, Florida, 33315
10	Gabbey	Cleever	7775802664	152 Cascade Terre Haute, Indiana, 47812

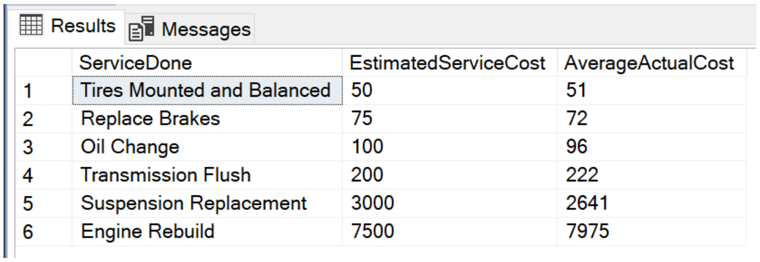
```
CREATE or ALTER VIEW ServicesProvided as
SELECT
       ServiceDone
       ,EstimatedDays
       ,EstimatedServiceCost
FROM CarService
GO

SELECT * FROM ServicesProvided
ORDER BY EstimatedDays
GO

-- Results:
```



	ServiceDone	EstimatedDays	EstimatedServiceCost
1	Oil Change	1	100
2	Replace Brakes	1	75
3	Transmission Flush	2	200
4	Tires Mounted and Balanced	2	50
5	Suspension Replacement	7	3000
6	Engine Rebuild	15	7500

```
--Showing us our prices and whether our estimates are accurate --
CREATE or ALTER VIEW CompareCosts as
SELECT
       CarService.ServiceDone
       ,CarService.EstimatedServiceCost
       ,Avg(CarCarService.Actualcost) As AverageActualCost
FROM CarCarService
       Join CarService on CarService.CarServiceID = CarCarService.ServiceID
Group by ServiceDone, EstimatedServiceCost
GO
```

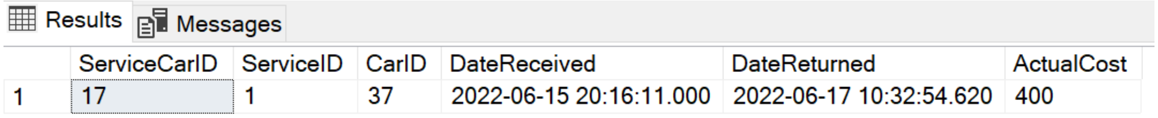
```
SELECT * FROM CompareCosts
GO

-- Results:
```



	ServiceDone	EstimatedServiceCost	AverageActualCost
1	Tires Mounted and Balanced	50	51
2	Replace Brakes	75	72
3	Oil Change	100	96
4	Transmission Flush	200	222
5	Suspension Replacement	3000	2641
6	Engine Rebuild	7500	7975

Programming Objects

```
--Complete a job in the system --

CREATE Or ALTER PROCEDURE FinishJob(@ServiceCarID int, @cost as int) AS
BEGIN
       UPDATE CarCarService SET DateReturned = GETDATE(), ActualCost = @cost
       WHERE ServiceCarID = @ServiceCarID
END
GO

DECLARE @new int
SET @new = 17
EXEC FinishJob @new, 400
SELECT * FROM CarCarService where ServiceCarID = @new
```

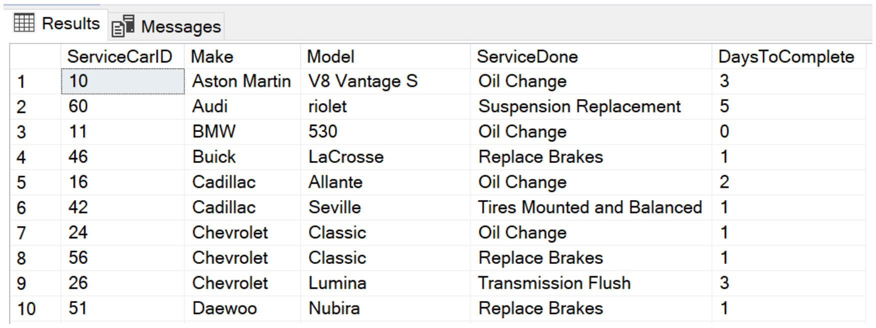


	ServiceCarID	ServiceID	CarID	DateReceived	DateReturned	ActualCost
1	17	1	37	2022-06-15 20:16:11.000	2022-06-17 10:32:54.620	400

```
--Check what cars got service done and the days to complete --

CREATE OR ALTER FUNCTION dbo.deadlines(@CarCarServiceID int)
RETURNS INT AS
BEGIN
       DECLARE @returnvalue varchar(20)
       DECLARE @received datetime
       DECLARE @returned datetime
       SELECT  @received = DateReceived, @returned = DateReturned
       FROM CarCarService
       JOIN Car on Car.CarID = CarCarService.CarID
       JOIN CarService on CarServiceID = CarCarService.ServiceID
       WHERE CarCarService.ServiceCarID = @CarCarServiceID
       SET @returnvalue = datediff(dd,@received,@returned)
       RETURN @returnvalue
END
GO

SELECT
       ServiceCarID
       ,Make
       ,Model
       ,ServiceDone
       ,dbo.deadlines(ServiceCarID) as DaysToComplete
```

```
        FROM CarCarService
        JOIN Car on Car.CarID = CarCarService.CarID
        JOIN CarService on CarServiceID = CarCarService.ServiceID
        Order by Make, Model, ServiceDone
GO
```

```
--Results:
```

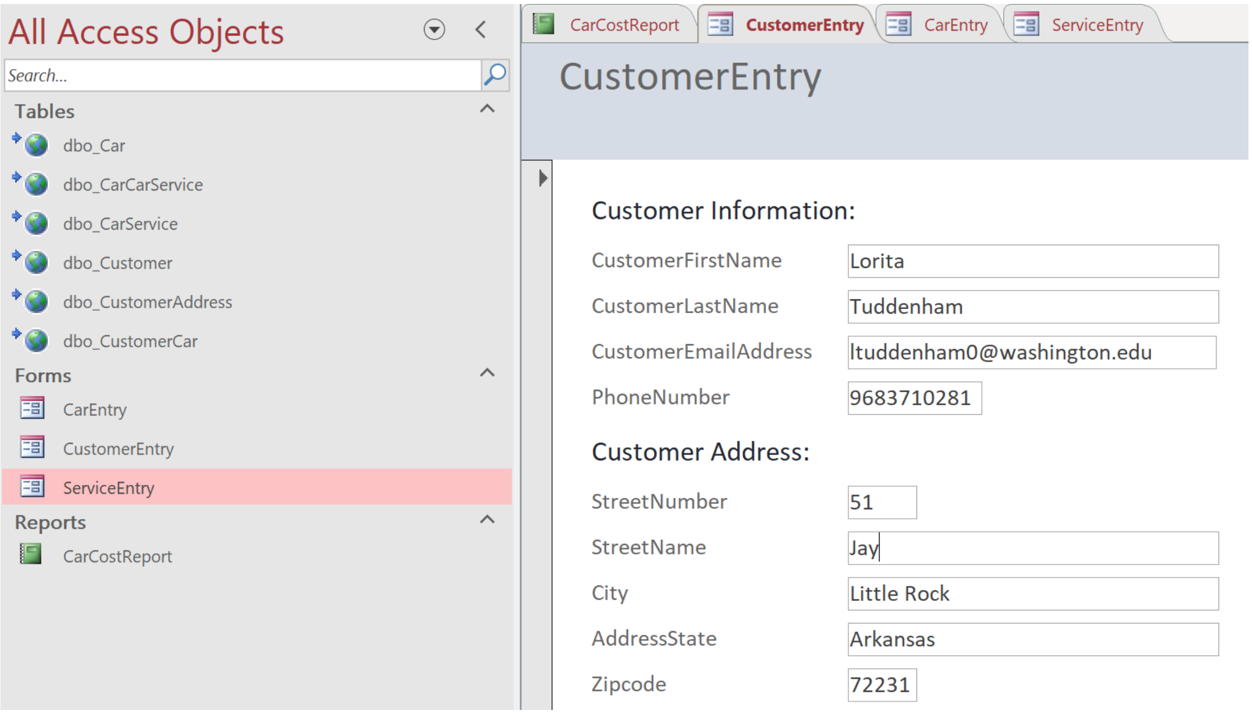


	ServiceCarID	Make	Model	ServiceDone	DaysToComplete
1	10	Aston Martin	V8 Vantage S	Oil Change	3
2	60	Audi	riolet	Suspension Replacement	5
3	11	BMW	530	Oil Change	0
4	46	Buick	LaCrosse	Replace Brakes	1
5	16	Cadillac	Allante	Oil Change	2
6	42	Cadillac	Seville	Tires Mounted and Balanced	1
7	24	Chevrolet	Classic	Oil Change	1
8	56	Chevrolet	Classic	Replace Brakes	1
9	26	Chevrolet	Lumina	Transmission Flush	3
10	51	Daewoo	Nubira	Replace Brakes	1

User Interface

Using Access below to enter in data and report financial data.

Customer Entry Form



CustomerEntry

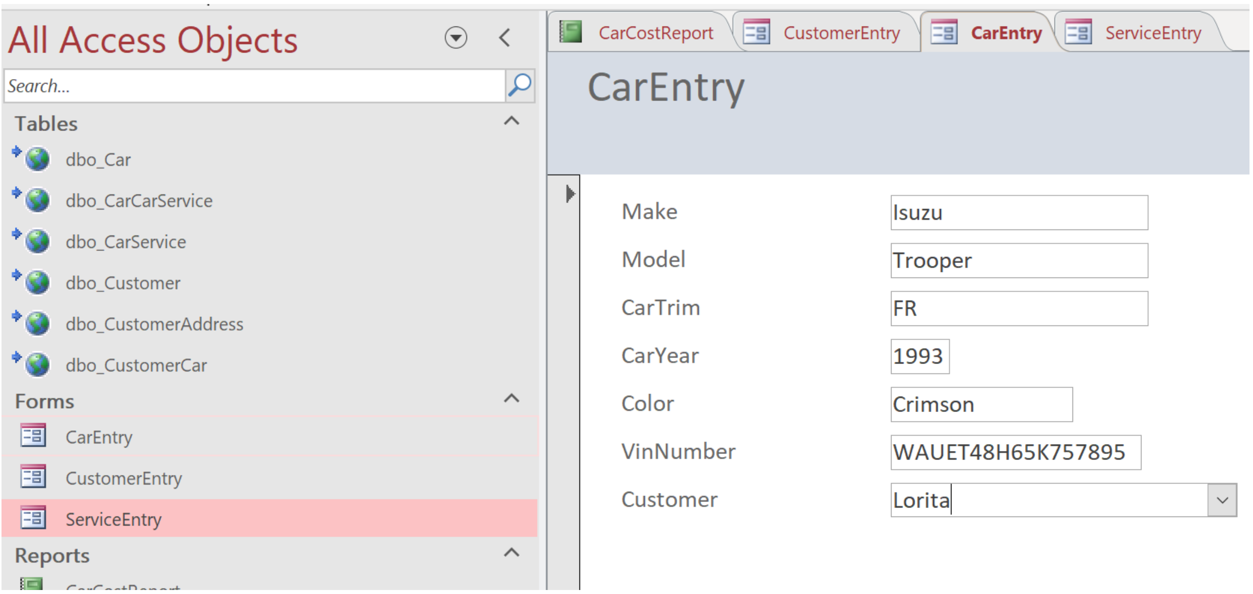
Customer Information:

CustomerFirstName	Lorita
CustomerLastName	Tuddenham
CustomerEmailAddress	ltuddenham0@washington.edu
PhoneNumber	9683710281

Customer Address:

StreetNumber	51
StreetName	Jay
City	Little Rock
AddressState	Arkansas
Zipcode	72231

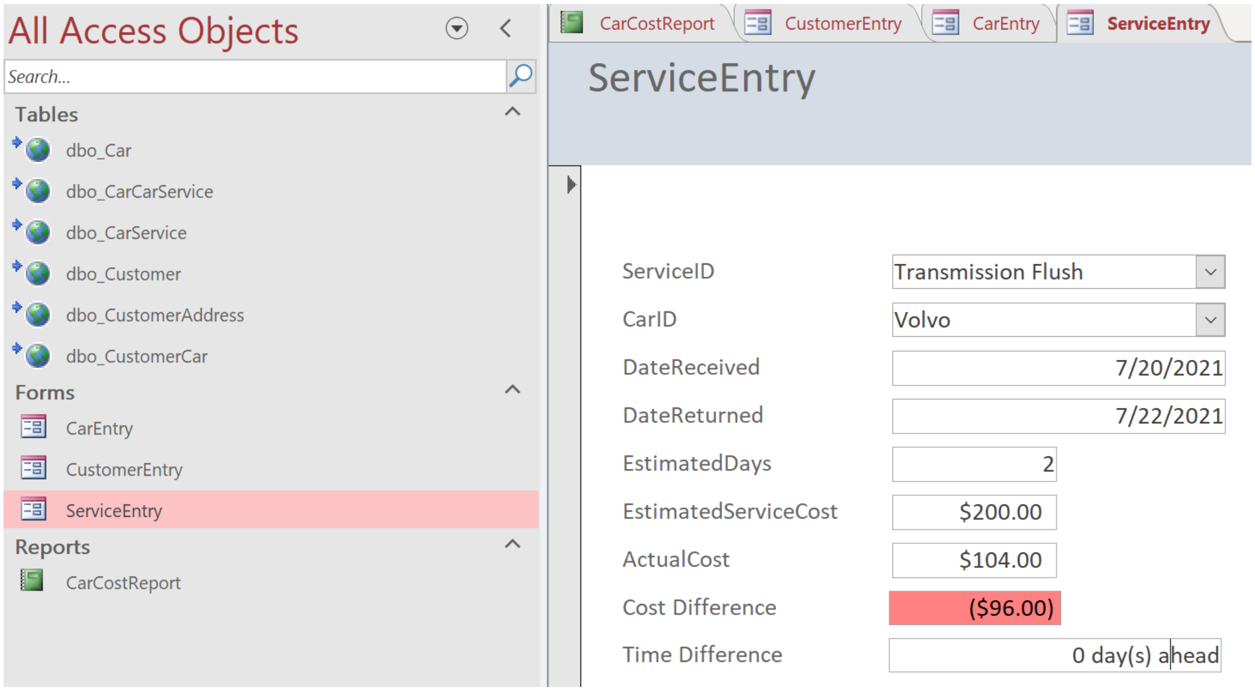
Richard Sangster
IST659 M407 Project
Spring 2022
Car Entry Form



CarEntry

Make	Isuzu
Model	Trooper
CarTrim	FR
CarYear	1993
Color	Crimson
VinNumber	WAUET48H65K757895
Customer	Lorita

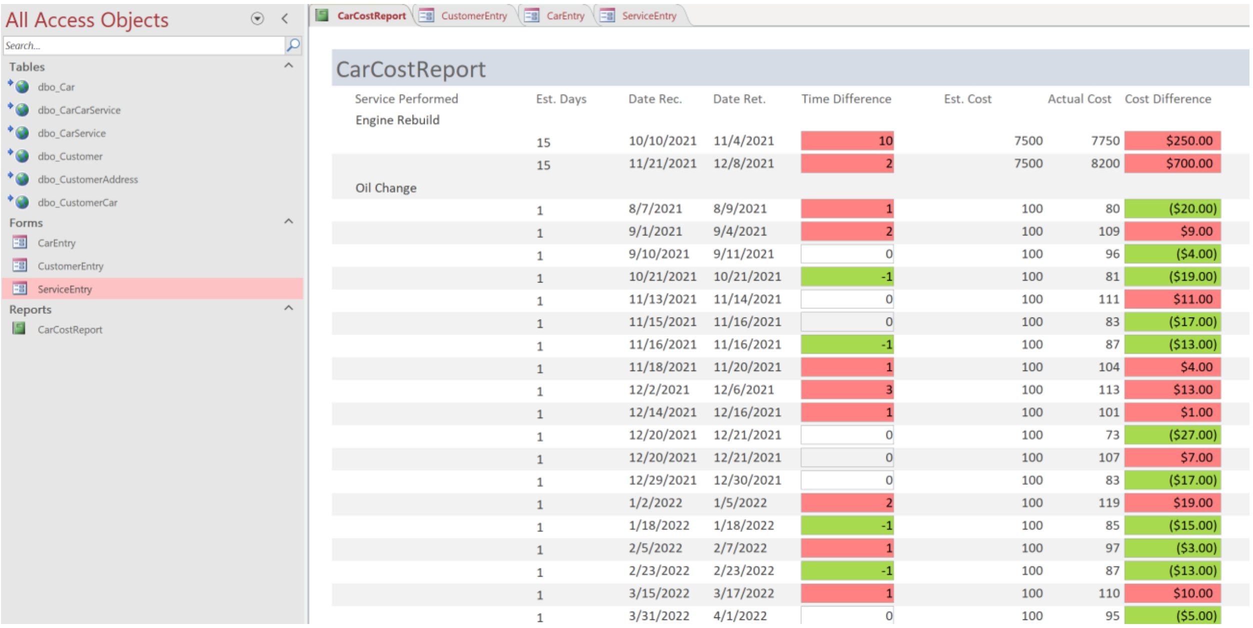
Service for cars entry



ServiceEntry

ServiceID	Transmission Flush
CarID	Volvo
DateReceived	7/20/2021
DateReturned	7/22/2021
EstimatedDays	2
EstimatedServiceCost	$200.00
ActualCost	$104.00
Cost Difference	($96.00)
Time Difference	0 day(s) ahead

Richard Sangster
IST659 M407 Project
Spring 2022
Report of car services (estimates vs. actual)



CarCostReport

Service Performed	Est. Days	Date Rec.	Date Ret.	Time Difference	Est. Cost	Actual Cost	Cost Difference
Engine Rebuild							
	15	10/10/2021	11/4/2021	10	7500	7750	$250.00
	15	11/21/2021	12/8/2021	2	7500	8200	$700.00
Oil Change							
	1	8/7/2021	8/9/2021	1	100	80	($20.00)
	1	9/1/2021	9/4/2021	2	100	109	$9.00
	1	9/10/2021	9/11/2021	0	100	96	($4.00)
	1	10/21/2021	10/21/2021	-1	100	81	($19.00)
	1	11/13/2021	11/14/2021	0	100	111	$11.00
	1	11/15/2021	11/16/2021	0	100	83	($17.00)
	1	11/16/2021	11/16/2021	-1	100	87	($13.00)
	1	11/18/2021	11/20/2021	1	100	104	$4.00
	1	12/2/2021	12/6/2021	3	100	113	$13.00
	1	12/14/2021	12/16/2021	1	100	101	$1.00
	1	12/20/2021	12/21/2021	0	100	73	($27.00)
	1	12/20/2021	12/21/2021	0	100	107	$7.00
	1	12/29/2021	12/30/2021	0	100	83	($17.00)
	1	1/2/2022	1/5/2022	2	100	119	$19.00
	1	1/18/2022	1/18/2022	-1	100	85	($15.00)
	1	2/5/2022	2/7/2022	1	100	97	($3.00)
	1	2/23/2022	2/23/2022	-1	100	87	($13.00)
	1	3/15/2022	3/17/2022	1	100	110	$10.00
	1	3/31/2022	4/1/2022	0	100	95	($5.00)

Reflection

The next time you go through the process of creating a database, what will you do differently now that you have been through the whole process?

I think the next time I make a database I will think more about how the database will work. I think the logistic model helped to transform the concept of the database but I needed to tweak the database even after I started to create it. I think the main takeaway is that the database needs to be flexible to start because there are new features that you may want or need to add as you build it. I also think that I had a big focus on the code to perform some of the data manipulation but there are easier tools in other software's (i.e. access) which can perform the same tasks automatically. Looking back there is a lot of steps to make each segment as well and I think it is important to take each table step by step and connect it all together. I would say picking a good starting point is important because that will dictate how smoothly the building process goes.