Table of Contents

Phase I

1. FACT FINDING AND INFORMATION GATHERING........................................5
   1.1. INTRODUCTION TO ENTERPRISE/ORGANIZATION...............................5
   1.2. DESCRIPTION OF FACT-FINDING TECHNIQUES....................................5
   1.3. THE PART OF ENTERPRISE YOU ARE DESIGNING CONCEPTUAL DATABASE..........................................................5
   1.4. ITEMIZED DESCRIPTIONS OF ENTITY SETS AND RELATIONSHIP SET........6
   1.5. USER GROUPS, DATA VIEWS AND OPERATIONS..................................6

2. CONCEPTUAL DATABASE DESIGN.............................................................7
   2.1. ENTITY SET DESCRIPTION....................................................................7-12
   2.2. RELATIONSHIP SET DESCRIPTION.....................................................12-13
   2.3. RELATED ENTITY SET .........................................................................13
   2.4. E-R DIAGRAM .....................................................................................14

Phase II

3. CONCEPTUAL DATABASE AND LOGICAL DATABASE ..............................15
   3.1. HISTORY AND EXPLANATION OF ER AND RELATIONAL MODELS..........15
       3.1.1 - DESCRIPTION ...........................................................................15
       3.1.2 - COMPARISON ............................................................................15
       3.1.3 – CONVERSION .............................................................................15-16
   3.2. CONSTRAINTS ......................................................................................16

4. CONVERTING E-R DATABASE INTO A RELATIONAL DATABASE .............16
   4.1 - RELATION SCHEMA ..........................................................................16 - 20
   4.2 - SAMPLE DATA OF RELATION .............................................................20 - 23

5. SAMPLE QUERIES ....................................................................................23
   5.1 - DESIGN OF QUERIES .........................................................................23-24
   5.2. RELATIONAL ALGEBRA, TUPLE RELATIONAL CALCULUS AND DOMAIN RELATIONAL CALCULUS OF QUERIES .................................................................24-28
13.4 - PACKAGE AND ITS SYNTAX .......................................................... 50-51
13.5 - TRIGGER AND ITS SYNTAX .................................................... 51-52

14 - ORACLE PL/SQL SUBPROGRAM .................................................. 52

14.1 - STORED PROCEDURES ............................................................. 53
14.1.1 – JGDM_NEWCUSTOMER ....................................................... 53-54
14.1.2 – JGDM_NEWSERVICETYPE ............................................... 54
14.1.3 – JGDM_NEWVEHICLES ......................................................... 54-55
14.1.4 – JGDM_DELETECUSTOMER ............................................... 55-56
14.1.5 – JGDM_DELETESERVICETYPE ............................................. 56
14.1.6 – JGDM_DELETEVEHICLES ................................................... 56-56
14.1.6 – JGDM_DELETEEMPLOYEE ............................................... 56
14.2 - TRIGGERS .............................................................................. 57
14.2.1 – JGDM_CUSTOMER_TRIGGER ................................................ 57
14.2.2 – JGDM_SERVICETYPE_TRIGGER ......................................... 57
14.2.3 – JGDM_VEHICLES_TRIGGER ................................................ 57
14.2.4 – JGDM_EMPLOYEE_TRIGGER ............................................... 58
14.2.5 – JGDM_MANUFACTURER_TRIGGER ........................................ 58

Phase V

15 – DAILY ACTIVITIES OF THE USER GROUP ..................................... 59
16 – RELATIONS, VIEWS, AND SUBPROGRAMS .................................. 59-61
17 – SCREEN SHOTS OF THE PROGRAM ............................................ 61-65
18 – DESCRIPTION OF PROGRAM AND CODE .................................. 65-66
19 – STEPS OF DESIGNING AND CREATING A DATABASE ................. 66-67
20 – ENBEDDED QUESTIONS ............................................................. 67-68
Phase I

1. FACT FINDING TECHNIQUES AND INFORMATION GATHERING

1.1 Introduction & Description of Enterprise

The business that will be used to model a database for this project will be Toyota manufacturing; this database will consist of the new brands of cars and insert the different manufactured car that goes through Toyota. The basic activities of the business includes getting the cars/trucks from the manufacture, put them on the lot, sell the vehicles to customers, record the data, and restart the cycle each time keeping track of our sales.

1.2 Facts & Techniques

In order to generate the database knowledge with the types of entities that are within the database will have to be clear. Contacting Bill Wrights Toyota branch would be a great aspect in our research. Discussing the types of things that are within the database with others in the field would greatly benefit us, and give an advantage.

1.3 Main Database design

The main ideal description that our database will partake in is selling only our new cars to customers. The database will store the many different types of cars we have in stock that are only new, along will the records of which employee sold the car/truck, whom he may have sold the car/truck too, how much was the car/truck sold for and how many they may have sold. Some of the main entities that are going to be the bases of our database will be “Vehicles”, “Trucks and Cars” that are in stock, Employee’s, Customers, and Services. The Car Vehicles will hold the information for the Cars and Tucks we hold, in that the different types of Cars/Trucks varies between the actual types such as SUV’s, 2D, 4D, Pick-ups and 4WD. Which then connects the customer and their interest in the vehicles. The customer then identifies what type of vehicle they are interested in and inform the employee of the Dealership that they're interested in a vehicle. The Employee then discusses matters with the customers and eventually convinces the customer to purchase a vehicle. Before the vehicle is purchased, the employee and customer’s go through the sales papers, so the customer’s data can be saved and sent off to the main database.
1.4 Entities and Relationship Description

The primary entity of the database in which the user will interact most with is being the Customer, they bring there basic information is going to be needed when they are in the process buying a car. Also, the Vehicles because it comes with many interactions. It comes with all of the major aspects that the user will need in order to find the car their looking for. As an example when the user is looking into finding a car, the car Vehicles brings up the option for a “Truck” or a “Car” which has the attributes SUV, Pickup, and 4WD for the truck, and 2D and 4D for the car’s attributes. Within it all is the attributes that apply to the selection of the vehicle such as, color, mileage, Model, year, etc.

In addition to the Car Vehicles is the next step in the process of buying a car is contacting our Dealership with the provided information on the database to check when the car selected by the user will be available. Included in these attributes is the basic information is the “address”, “Phone #”.

Afterwards you contact the sales department within our Dealership in order to schedule a time and date, and find out the basic attributes, commission and rate.

After contacting everyone and scheduling a time and date you meet with one of the employees at the dealership, this is when the user finds out the First and Last Name, Position, and basic information of the employee that helps the user. The employee eventually makes the deal with the user and goes through the Sales Paper, included in that entity is the attributes of the sales date, sale price, etc.

1.5 User Groups, Data views and Operation

The main groups to have access to the database will strictly be the sales departments, the Dealership of the Toyota dealership we are working on selling cars for, and finally the employee who actually sales the car to the customer. The only entities that'll be accessible is the car Vehicles, which includes the list of new trucks, which vary between SUV’s, Pickups, and 4WD; and also cars which varies between 2 Doors and 4 Doors. The employee will be the sales representative and keep a listing of how many cars and what type was sold. It will then go through the sales department of our Dealership, where it will be kept and stored then sent to the main head quarters of Toyota.
2. CONCEPTUAL DATABASE DESIGN

2.1 Entity Set Description

2.1.1 Entity Manufacturer

- Name: Manufacturer
- Description: The entity Manufacturer is the Company that brings the cars to us and the company from which we order parts and cars from as well.
- Attributes:

<table>
<thead>
<tr>
<th>Name</th>
<th>ManufacturerID</th>
<th>MName</th>
<th>MAddress</th>
<th>MPhone</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description</td>
<td>Manufacturer’s ID number</td>
<td>Manufacturers Name</td>
<td>Manufacturer’s Address</td>
<td>Manufacturer’s Phone</td>
</tr>
<tr>
<td>Domain/Type</td>
<td>Integer</td>
<td>String</td>
<td>String</td>
<td>Integer String</td>
</tr>
<tr>
<td>Value Range</td>
<td>Max ID +1</td>
<td>ANY</td>
<td>ANY</td>
<td>10</td>
</tr>
<tr>
<td>Default Value?</td>
<td>NO</td>
<td>NO</td>
<td>NO</td>
<td>NO</td>
</tr>
<tr>
<td>Null Allowed?</td>
<td>NO</td>
<td>NO</td>
<td>NO</td>
<td>NO</td>
</tr>
<tr>
<td>Unique</td>
<td>YES</td>
<td>NO</td>
<td>NO</td>
<td>YES</td>
</tr>
<tr>
<td>Single or Multi-Value?</td>
<td>Single</td>
<td>Single</td>
<td>Multi</td>
<td>Single</td>
</tr>
<tr>
<td>Simple or Composite?</td>
<td>Simple</td>
<td>Simple</td>
<td>Composite</td>
<td>Simple</td>
</tr>
</tbody>
</table>

- Candidate Keys: Manufacturer_ID, Name
- Primary Key: Manufacturer_ID
- Weak/Strong: Strong
- Fields to be Indexed: Manufacturer_ID, Name
2.1.2 Entity Vehicles:

- Name: Vehicles
- Description: The Vehicles entity will store vehicles description. Each vehicle will have a: year, make, model, VIN, quantity, color, Acquired Date, Acquired Price, Sale Date, Dealership ID, and Vendor ID. The VIN number will be a 12 digit unique. Quantity will show how many cars of that same year, make and, model are in stock. Acquired date will show when the car was originally purchased from the vendor, and acquired price will show how much was paid to the vendor. The Dealership ID will identify the Car dealership that purchased the vehicle. The Vendor ID represents the Company that manufactured the vehicle and who also sold it to the Dealership. Sale Date will show the date in which the car was sold to a customer.
- Attributes:

<table>
<thead>
<tr>
<th>Name</th>
<th>Vehicle_ID</th>
<th>VIN</th>
<th>Miles</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description</td>
<td>Vehicle’s ID number at the dealership</td>
<td>DMV’s Vehicle ID</td>
<td>Mileage on Car</td>
</tr>
<tr>
<td>Domain/Type</td>
<td>Integer</td>
<td>String</td>
<td>String</td>
</tr>
<tr>
<td>Value-Range</td>
<td>0 to max</td>
<td>ANY</td>
<td>Any</td>
</tr>
<tr>
<td>Default Value</td>
<td>Max ID +1</td>
<td>No</td>
<td>NO</td>
</tr>
<tr>
<td>Null?</td>
<td>No</td>
<td>NO</td>
<td>NO</td>
</tr>
<tr>
<td>Unique?</td>
<td>YES</td>
<td>NO</td>
<td>YES</td>
</tr>
<tr>
<td>Single or Multiple Value</td>
<td>Single</td>
<td>Single</td>
<td>Single</td>
</tr>
<tr>
<td>Simple or Composite</td>
<td>Simple</td>
<td>Simple</td>
<td>Simple</td>
</tr>
</tbody>
</table>
• Candidate Key: VehicleID, VIN,
• Primary Key: Vehicle_ID
• Weak/Strong: Strong
• Fields to be Indexed: VehicleID, VIN,

2.1.3. Entity Vehicle Type

• Name: Vehicle Type
• Description: The entity vehicle type describes what type of car is the vehicle.
• Attributes:

<table>
<thead>
<tr>
<th>Name</th>
<th>Model_id</th>
<th>Year</th>
<th>Model</th>
<th>Type</th>
<th>MSRP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description</td>
<td>Model Identifier</td>
<td>Manufactured Year</td>
<td>Name of Car</td>
<td>Kind of vehicle</td>
<td>Manufacturer’s Suggested Price</td>
</tr>
<tr>
<td>Domain/Type</td>
<td>String</td>
<td>INT</td>
<td>String</td>
<td>String</td>
<td>Float</td>
</tr>
<tr>
<td>Value – Range</td>
<td>ANY</td>
<td>INT &gt;= 2013</td>
<td>ANY</td>
<td>ANY</td>
<td>ANY</td>
</tr>
<tr>
<td>Default Value</td>
<td>Model_Year</td>
<td>NO</td>
<td>NO</td>
<td>NO</td>
<td>NO</td>
</tr>
<tr>
<td>Null Allowed?</td>
<td>NO</td>
<td>NO</td>
<td>NO</td>
<td>NO</td>
<td>NO</td>
</tr>
<tr>
<td>Unique?</td>
<td>YES</td>
<td>NO</td>
<td>NO</td>
<td>NO</td>
<td>NO</td>
</tr>
<tr>
<td>Single or Multi-Value?</td>
<td>Single</td>
<td>Single</td>
<td>Single</td>
<td>Single</td>
<td>Single</td>
</tr>
<tr>
<td>Simple or Composite?</td>
<td>Simple</td>
<td>Simple</td>
<td>Simple</td>
<td>Simple</td>
<td>Simple</td>
</tr>
</tbody>
</table>

• Candidate Key: Model_id
• Primary Key: Model_id
• Weak/Strong: Strong
• Fields to be Indexed: Model_id
2.1.4. Entity Employees

- Name: Employees
- Description: This entity will store every employee’s information. Each employee record will contain: Names, position, Email, phone numbers, and address. This entity however is only for those employed by the sales department of the Dealership.
- Attributes:

<table>
<thead>
<tr>
<th>Name</th>
<th>EName</th>
<th>EmployeeID</th>
<th>Position</th>
<th>EAddress</th>
<th>Hphone</th>
<th>Cphone</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description</td>
<td>Employee’s Name</td>
<td>Employee’s ID Number</td>
<td>Job Position in the dept.</td>
<td>Employee’s home address</td>
<td>Employee’s Home Phone</td>
<td>Employee’s Cell phone</td>
</tr>
<tr>
<td>Domain/Type</td>
<td>String</td>
<td>Integer</td>
<td>String</td>
<td>String</td>
<td>Integer</td>
<td>Integer</td>
</tr>
<tr>
<td>Value – Range</td>
<td>ASCII</td>
<td>MaxID+1</td>
<td>ANY</td>
<td>ANY</td>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td>Default Value</td>
<td>NO</td>
<td>NO</td>
<td>Salesman</td>
<td>NO</td>
<td>NO</td>
<td>NO</td>
</tr>
<tr>
<td>Null Allowed?</td>
<td>NO</td>
<td>NO</td>
<td>NO</td>
<td>NO</td>
<td>YES</td>
<td>NO</td>
</tr>
<tr>
<td>Unique?</td>
<td>NO</td>
<td>YES</td>
<td>NO</td>
<td>NO</td>
<td>YES</td>
<td>YES</td>
</tr>
<tr>
<td>Single or Multi-Value?</td>
<td>Single</td>
<td>Single</td>
<td>Single</td>
<td>Multi-value</td>
<td>Single</td>
<td>Single</td>
</tr>
<tr>
<td>Simple or Composite?</td>
<td>Simple</td>
<td>Simple</td>
<td>Simple</td>
<td>Composite</td>
<td>Simple</td>
<td>Simple</td>
</tr>
</tbody>
</table>

- Candidate Key: EID
- Primary Key: EID
- Weak/Strong: Strong
- Field to be Indexed: EID

2.1.5. Entity Customer

- Name: Customer
• Description: This Entity hold information on the customer who purchased the vehicle.

• Attributes:

<table>
<thead>
<tr>
<th>Name</th>
<th>Customer_id</th>
<th>CFirstName</th>
<th>CLastName</th>
<th>CAddress</th>
<th>CPhone</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description</td>
<td>Customer’s Identification in database</td>
<td>Customer’s FirstName</td>
<td>Customer’s LastName</td>
<td>Customer’s Address</td>
<td>Customer’s Phone Number</td>
</tr>
<tr>
<td>Domain/Type</td>
<td>Integer</td>
<td>String</td>
<td>String</td>
<td>String</td>
<td>Integer String</td>
</tr>
<tr>
<td>Value – Range</td>
<td>ANY</td>
<td>ANY</td>
<td>ANY</td>
<td>ANY</td>
<td>10</td>
</tr>
<tr>
<td>Default Value</td>
<td>NO</td>
<td>NO</td>
<td>NO</td>
<td>NO</td>
<td>NO</td>
</tr>
<tr>
<td>Null Allowed?</td>
<td>NO</td>
<td>NO</td>
<td>NO</td>
<td>NO</td>
<td>NO</td>
</tr>
<tr>
<td>Unique?</td>
<td>YES</td>
<td>NO</td>
<td>NO</td>
<td>NO</td>
<td>YES</td>
</tr>
<tr>
<td>Single or Multi-Value?</td>
<td>Single</td>
<td>Single</td>
<td>Single</td>
<td>Single</td>
<td>Single</td>
</tr>
<tr>
<td>Simple or Composite?</td>
<td>Simple</td>
<td>Simple</td>
<td>Simple</td>
<td>Composite</td>
<td>Simple</td>
</tr>
</tbody>
</table>

• Candidate Key: Customer_id,
• Primary Key: Customer_id
• Weak/Strong: Strong
• Fields to be Indexed: Customer_id

2.1.6. Entity Service Type

• Name: Service Type
• Description: The entity Service Type represents the service the vehicle has received during its duration in the dealership.

• Attributes:

<table>
<thead>
<tr>
<th>Name</th>
<th>Service_id</th>
<th>Service Name</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description</td>
<td>Service Identification #</td>
<td>Service Performed</td>
<td>Result of inspection</td>
</tr>
<tr>
<td>Domain/Type</td>
<td>String</td>
<td>String</td>
<td>String</td>
</tr>
<tr>
<td>Value-Range</td>
<td>ANY</td>
<td>ANY</td>
<td>ANY</td>
</tr>
<tr>
<td>Default Value</td>
<td>YES</td>
<td>NO</td>
<td>NO</td>
</tr>
</tbody>
</table>
### Null Allowed?

<table>
<thead>
<tr>
<th>Null Allowed?</th>
<th>NO</th>
<th>NO</th>
<th>NO</th>
</tr>
</thead>
</table>

### Unique?

<table>
<thead>
<tr>
<th>Unique?</th>
<th>NO</th>
<th>NO</th>
<th>NO</th>
</tr>
</thead>
</table>

### Single or Multi-Value?

<table>
<thead>
<tr>
<th>Single or Multi-Value?</th>
<th>Single</th>
<th>Single</th>
<th>Single</th>
</tr>
</thead>
</table>

### Simple or Composite

<table>
<thead>
<tr>
<th>Simple or Composite</th>
<th>Simple</th>
<th>Simple</th>
<th>Simple</th>
</tr>
</thead>
</table>

- Candidate Key: Service_id
- Primary Key: Service_id
- Weak/Strong: Strong
- Field to be Indexed: Service_id

---

### 2.2 Relationship Set Description

**Relationship ServicedBy**

- **Name ServicedBy:**
- **Description:** The serviced relationship is a relation between the Vehicles entity, ServiceType entity, and Employee entity. It describes which Employee serviced what Vehicle for Maintenance.
- **Entity Set involved is Employee, ServiceType, and Vehicles**
- **Cardinality is Employee 1..* ServiceType and Vehicles 1.. * ServiceType**
- **Participation:** Total/Mandatory

**Relationship Purchase:**

- **Name:** Purchased
- **Description:** The Purchased relationship is a relation between the Manufacturer entity and Vehicles entity. It describes which vehicles did the dealership buy from which manufacturer.
- **Entity Set involved is Manufacturer and Vehicles.**
- **Cardinality is *..*
• Participation: Total

Relationship Sales:

• Name: Sales
• Description: The Sales relationship is a relation between the Customer entity, Employee entity, and Vehicles entity. It describes the vehicle, customer, and employee that was involved in the sale.
• Entity set involved is Vehicles, Customer, and Employee.
• Cardinality is Customer 1..*Vehicles and Employee 1.. *Vehicles
• Participation: Total

Relationship IS:

• Name: Is
• Description: The IS relationship is a relation between the Vehicles and the Vehicle Type entities. It describes what type of vehicle was purchased on the sale.
• Entity set involved is Vehicles and Vehicle Type
• Cardinality is *..1
• Participation: Total/Mandatory

2.3 Related Entity Set

Vehicles

This entity is generalized because it can be broken down to subclasses of Trucks or cars. For which we might add a disjoint constraint.
2.4 ER Diagram
Phase II

3. CONCEPTUAL DATABASE AND LOGICAL DATABASE

3.1 History and Explanation of ER & Relational Models

3.1.1 Description

An Entity-Relationship model is used during the breakdown to present the information of a database. The ER Model is based on multiple things such as algebra, theory, logic, etc. It is also one of the most important steps needed in software engineering and informational database design. The object that is the base of the ER model represents higher power database models. The descriptive value of this model describes each relation, entity, and attribute in the model. It is used in the design of theoretical schemas for database applications. Schemas used in database applications are viewed for graphical alteration that is known as the ER Model.

The Relational Model was invented by Dr. Edgar F. Codd and popularized in the 1970’s and used as a general model of data. The model is normally used due the simplicity and popularized mathematical background. The model is seen as a declarative concept in making data and querying it. The model represents a collective database of relations. Every row established in each table signifies a collection of connected data. Each value has a own specific name such as a row, it is referred to as a tuple; the header is the attributes; and the table is referred to as the relation, with the data types relating each value in each column is exemplified by a domain of possible values.

3.1.2 Comparing

The ER and Relational Model differ from the ER Model using math relations to express the relationships between entities, and Relational Models uses the Cartesian product. They both have similar traits; ER model is expressed as a tuple of entities. In the relational model, a Cartesian product of data areas is a relation, while in the ER model a Cartesian product of entities is a shown in relationships. The relational model construct is used to express the structured data values while in the ER model the same construct is used to express the structure of the entities.

3.1.3 Conversion

A weak entity can never exist while it’s alone, the primary key of it are made of the foreign key. The simple entity is divided into small subparts and used to describe its component. When conversions come into play they can determine whether the value is single or multi-
valued, an example would be the degree of weather. Leading to the different types of relationships there are types such as:

- one-to-one, it makes a table that has columns for each entity set.
- many-to-one and one-to-many, have columns for numerous adjacent entity set and column for one side entity set's primary key.
- Union, builds the table with as many columns as there are attributes for the union of the primary keys of all participating entity sets.

The union uses all primary keys of the entity sets that are on many sides. If there are more than two entities than expected, the new step is to create a table for each subclass entity set include all attribute of that subclass entity and attributes of super class entity set. Recursive relationship connects a single class type to itself.

3.2 Constraints

The value that the primary key holds is to identify tuples in the relation, more often presented by highlighting. Referential Integrity constraint is clearly stated within the relations placed to maintain reliability that the tuples have between the relations. When a key is seen as if it refers to different relations, it is mentioned to as the entities foreign key; which must be present before making a relation. In order for constraints to be directly expressed in schemas, it must be enforced by the submissions databases.

4.1 Relation Schema

MANUFACTURER:

MANUFACTURER (Manufacturer_id, Name, Address, City, St, Zip, Phone)

1. Manufacturer_id – Domain: Integer
a. Default Value: (Max Client_id + 1)
b. Primary Key (Unique and NOT NULL)

2. Name – Domain: Varchar (50)
   a. NOT NULL
   b. Unique

3. Address - Domain: Varchar (50)
   a. NOT NULL

4. City - Domain: Varchar (20)
   a. NOT NULL

5. St - Domain: Char (2)
   a. NOT NULL

6. Zip - Domain: Integer
   a. NOT NULL

7. Phone - Domain: Integer

Candidate Keys:

1. Manufacturer_id (Primary key)
2. Name (Unique)

VEHICLE TYPE

VEHICLETYP(Model_id, Year, Type, MSRP)

1. Model_id – Domain: Varchar(15)
   a. Default Value: Model Name followed by the Year
   b. Primary(Unique and Not Null)
2. Year – INT
   a. Not Null
3. Type – Varchar(10)
   a. Not Null
4. MSRP – Domain: float
   a. Not Null

Candidate Keys: Model_id

VEHICLES:

VEHICLES (Vehicles_id, VIN, Miles, Color, Model_id, Aq_Date, Aq.Price, Manufacturer_id, Service_id, Sale_Date, Sale_Price, Commission, Employee_id, Customer_id, )

1. Vehicles_id - Domain: Integer
   a. Default Value: (First two digits denote year) (3rd and fourth
digit denote month aquired) (last three digits increase by 1 for every
incoming vehicle for that month.

b. Primary Key (Unique and NOT NULL)

2. VIN - Domain varchar(30)
a. NOT NULL

5. Miles – Domain: Integer
   a. Default Value: int >= 0

6. Color – Domain: Varchar(10)
a. Not Null

7. Model_id – Varchar(20)
a. Not Null
   b. Foreign Key to VehicleType.Model_id

8. Aq_date - Domain: Date

9. Aq_price – float

10. Manufacturer_id – INT
    a. Not Null

11. Service_id - Domain: INT
    a. Foreign key to ServiceType.Service_id

12. Sale_Date – Domain: Date
    a. Null allowed

    a. Null Allowed


15. Employee_id – Domain: INT
    a. NULL ALLOWED
       b. Foreign key to Employee.Employee_id

    a. Null Allowed
       b. Customer_id

13. Employee_id – Domain: Integer
    a. Not Null
       b. Foreign Key to Employee.Employee_id

14. Manufacturer_id - Domain Integer
    a. NOT NULL
       b. Foreign Key to Manufacturer.Manufacturer_id

Candidate Keys:

1. Vehicle_id (Primary Key)
2. VIN

EMPLOYEE

EMPLOYEE(Employee_id, FirstName, LastName, Position, EAddress, EState, ECity, EZip, CPhone, HPhone)

1. Employee_id – Domain: Integer
a. Not Null  
b. Primary Key  
2. FirstName – Domain: Varchar(10)  
a. Not Null  
3. LastName – Domain: Varchar(10)  
4. Position – Domain: Varchar(20)  
5. EAddress – Domain: Varchar(50)  
6. EState – Domain: Char(2)  
a. Not Null  
7. EZip – Domain: Integer  
a. Not Null  
8. CPhone – Domain: Integer  
9. HPhone – Domain: Integer  
Candidate Keys: 
1. Employee_id (Primary Key)  

CUSTOMER:  

CUSTOMER(Customer_id, CFirstName, CLastName, CAddress, CCity, CState, CZip, Phone)  

1. Customer_id – Domain: Integer  
a. Default Value: (Max Customer_id + 1)  
b. Primary Key(Unique and Not Null)  
2. CFirstName- Domain: Varchar(10)  
3. CLastName – Domain: Varchar(10)  
4. CAddress – Domain: Varchar(50)  
a. Not Null  
5. CCity – Domain: Varchar(20)  
a. Not Null  
6. CState – Domain: Varchar(20)  
a. Not Null  
7. CZip – Domain: Integer  
a. Not Null  
8. Phone – Domain: Integer  

Candidate Keys:  
1. Customer_id (Primary Keys)  
2. CFirstName
SERVICETYPE

SERVICETYPE(Service_id, Service_Name, Service_date, Result, Vehicle_id, Employee_id)

1. Service_id – Domain: Varchar(20)
   a. Default value: (Vehicle_id)-(two digits increase by 1 for every service done).
   b. Primary (Not Null)
2. Service_Name – Domain: Varchar(20)
   a. Not Null
3. Service_date – Domain: Date
   a. Not Null
4. Result – Domain: Varchar(20)
   a. Not Null
5. Vehicle_id – Domain: INT
   a. Not Null
   b. Foreign Key to Vehicles.Vehicles_id
6. Employee_id – Domain: INT
   a. Not Null
   b. Foreign Key to Employee.Employee_id

Candidate Keys:
1. Service_id (Primary)

4.2 Sample Data of Relation

In this section, we will create and display relational instances for each table using excel. Later in the project we will insert this data into the database using SQL INSERT command.

MANUFACTURER(Manufacturer_id. Name, Address, City, State, Zip, Phone)

<table>
<thead>
<tr>
<th>Manufacturer_id</th>
<th>Name</th>
<th>Address</th>
<th>City</th>
<th>State</th>
<th>Zip</th>
<th>Phone</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Toyota Motor Manufacturing Mississippi, inc.</td>
<td>1200 Magnolia Dr.</td>
<td>Blue Springs</td>
<td>MS</td>
<td>38828</td>
<td>662 3173015</td>
</tr>
<tr>
<td>2</td>
<td>Toyota Motor Manufacturing Kentucky, Inc.</td>
<td>1001 Cherry Blossom Way</td>
<td>Georgetown</td>
<td>KY</td>
<td>40324</td>
<td>5028682000</td>
</tr>
<tr>
<td>3</td>
<td>Toyota Motor Manufacturing Texas, Inc</td>
<td>1 Lone Star Pass</td>
<td>San Antonio</td>
<td>TX</td>
<td>78264</td>
<td>2102634000</td>
</tr>
<tr>
<td>4</td>
<td>Toyota Motor Manufacturing Indiana, Inc</td>
<td>4000 Tulip Tree Dr</td>
<td>Princeton</td>
<td>IN</td>
<td>47670</td>
<td>8123872000</td>
</tr>
<tr>
<td>5</td>
<td>Toyota Motor Manufacturing Alabama, Inc</td>
<td>1 Cottonvalley Drive</td>
<td>Huntsville</td>
<td>AL</td>
<td>35810</td>
<td>2567465000</td>
</tr>
<tr>
<td>6</td>
<td>Toyota Motor Manufacturing West Virginia</td>
<td>1 Sugar Maple Ln</td>
<td>Buffalo</td>
<td>WV</td>
<td>25033</td>
<td>3049377000</td>
</tr>
</tbody>
</table>


<table>
<thead>
<tr>
<th>Model_id</th>
<th>Year</th>
<th>TypeofCar</th>
<th>MSRP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Camry13</td>
<td>2013</td>
<td>Car/4DR</td>
<td>21,531</td>
</tr>
<tr>
<td>Camry13</td>
<td>2013</td>
<td>Car/4DR</td>
<td>21,531</td>
</tr>
<tr>
<td>Camry13</td>
<td>2013</td>
<td>Car/4DR</td>
<td>21,531</td>
</tr>
<tr>
<td>Camry13</td>
<td>2013</td>
<td>Car/4DR</td>
<td>22,497</td>
</tr>
<tr>
<td>CamryHybrid13</td>
<td>2013</td>
<td>Car/4DR</td>
<td>25,302</td>
</tr>
<tr>
<td>CamryHybrid13</td>
<td>2013</td>
<td>Car/4DR</td>
<td>25,302</td>
</tr>
<tr>
<td>CamryHybrid13</td>
<td>2013</td>
<td>Car/4DR</td>
<td>25,302</td>
</tr>
<tr>
<td>Avalon13</td>
<td>2013</td>
<td>Car/4DR</td>
<td>29,214</td>
</tr>
<tr>
<td>Avalon13</td>
<td>2013</td>
<td>Car/4DR</td>
<td>29,214</td>
</tr>
<tr>
<td>AvalonHybrid13</td>
<td>2013</td>
<td>Car/4DR</td>
<td>33,396</td>
</tr>
<tr>
<td>AvalonHybrid13</td>
<td>2013</td>
<td>Car/4DR</td>
<td>33,396</td>
</tr>
<tr>
<td>Highlander13</td>
<td>2013</td>
<td>Truck/4X2</td>
<td>27,568</td>
</tr>
<tr>
<td>Highlander13</td>
<td>2013</td>
<td>Truck/4X2</td>
<td>27,568</td>
</tr>
<tr>
<td>Highlander13</td>
<td>2013</td>
<td>Truck/4X2</td>
<td>27,568</td>
</tr>
<tr>
<td>FJ Cruiser113</td>
<td>2013</td>
<td>Truck/4X2</td>
<td>26,707</td>
</tr>
<tr>
<td>FJ Cruiser213</td>
<td>2013</td>
<td>Truck/4X4</td>
<td>27,577</td>
</tr>
<tr>
<td>Sienna13</td>
<td>2013</td>
<td>Truck/FWD</td>
<td>25,905</td>
</tr>
<tr>
<td>Tacoma13</td>
<td>2013</td>
<td>Truck/4X2</td>
<td>18,726</td>
</tr>
<tr>
<td>Tacoma13</td>
<td>2013</td>
<td>Truck/4x2</td>
<td>20,352</td>
</tr>
<tr>
<td>Tundra13</td>
<td>2013</td>
<td>Truck/4x4</td>
<td>30,107</td>
</tr>
<tr>
<td>Prius13</td>
<td>2013</td>
<td>Car/4DR</td>
<td>19,273</td>
</tr>
<tr>
<td>Prius13</td>
<td>2013</td>
<td>Car/4DR</td>
<td>19,273</td>
</tr>
<tr>
<td>Prius13</td>
<td>2013</td>
<td>Car/4DR</td>
<td>19,273</td>
</tr>
</tbody>
</table>

**VEHICLETYPE (Model_id, Year, Type, MSRP)**

<table>
<thead>
<tr>
<th>Vehicles_id</th>
<th>VIN</th>
<th>Color</th>
<th>Model_id</th>
<th>Year</th>
<th>TypeofCar</th>
<th>MSRP</th>
</tr>
</thead>
<tbody>
<tr>
<td>1309001</td>
<td>JT152EAA100302159</td>
<td>Silver</td>
<td>Camry13</td>
<td>2013</td>
<td>Car/4DR</td>
<td>21,531</td>
</tr>
<tr>
<td>1309002</td>
<td>JT152EAA100302160</td>
<td>Black</td>
<td>Camry13</td>
<td>2013</td>
<td>Car/4DR</td>
<td>21,531</td>
</tr>
<tr>
<td>1309003</td>
<td>JT152EAA100302161</td>
<td>Black</td>
<td>Camry13</td>
<td>2013</td>
<td>Car/4DR</td>
<td>21,531</td>
</tr>
<tr>
<td>1309004</td>
<td>JT152EAA100302162</td>
<td>White</td>
<td>Camry13</td>
<td>2013</td>
<td>Car/4DR</td>
<td>22,497</td>
</tr>
<tr>
<td>1309005</td>
<td>JT152EAA100302163</td>
<td>Black</td>
<td>CamryHybrid13</td>
<td>2013</td>
<td>Car/4DR</td>
<td>25,302</td>
</tr>
<tr>
<td>1309006</td>
<td>JT152EAA100302164</td>
<td>White</td>
<td>CamryHybrid13</td>
<td>2013</td>
<td>Car/4DR</td>
<td>25,302</td>
</tr>
<tr>
<td>1309007</td>
<td>JT152EAA100302165</td>
<td>Green</td>
<td>CamryHybrid13</td>
<td>2013</td>
<td>Car/4DR</td>
<td>25,302</td>
</tr>
<tr>
<td>1309008</td>
<td>JT152EAA100302166</td>
<td>Green</td>
<td>Avalon13</td>
<td>2013</td>
<td>Car/4DR</td>
<td>29,214</td>
</tr>
<tr>
<td>1309009</td>
<td>JT152EAA100302167</td>
<td>Red</td>
<td>Avalon13</td>
<td>2013</td>
<td>Car/4DR</td>
<td>29,214</td>
</tr>
<tr>
<td>1309010</td>
<td>JT152EAA100302168</td>
<td>Silver</td>
<td>AvalonHybrid13</td>
<td>2013</td>
<td>Car/4DR</td>
<td>33,396</td>
</tr>
<tr>
<td>1309011</td>
<td>JT152EAA100302169</td>
<td>White</td>
<td>AvalonHybrid13</td>
<td>2013</td>
<td>Car/4DR</td>
<td>33,396</td>
</tr>
<tr>
<td>1309012</td>
<td>JT152EAA100302170</td>
<td>Blue</td>
<td>Highlander13</td>
<td>2013</td>
<td>Car/4DR</td>
<td>27,568</td>
</tr>
<tr>
<td>1309013</td>
<td>JT152EAA100302171</td>
<td>Black</td>
<td>Highlander13</td>
<td>2013</td>
<td>Car/4DR</td>
<td>27,568</td>
</tr>
<tr>
<td>1309014</td>
<td>JT152EAA100302172</td>
<td>Red</td>
<td>Highlander13</td>
<td>2013</td>
<td>Car/4DR</td>
<td>27,568</td>
</tr>
<tr>
<td>1309015</td>
<td>JT152EAA100302173</td>
<td>Silver</td>
<td>FJ Cruiser113</td>
<td>2013</td>
<td>Car/4DR</td>
<td>26,707</td>
</tr>
<tr>
<td>1309016</td>
<td>JT152EAA100302174</td>
<td>Black</td>
<td>FJ Cruiser213</td>
<td>2013</td>
<td>Car/4DR</td>
<td>27,577</td>
</tr>
<tr>
<td>1309017</td>
<td>JT152EAA100302175</td>
<td>Gold</td>
<td>Sienna13</td>
<td>2013</td>
<td>Car/4DR</td>
<td>25,905</td>
</tr>
<tr>
<td>1309018</td>
<td>JT152EAA100302176</td>
<td>White</td>
<td>Tacoma13</td>
<td>2013</td>
<td>Car/4DR</td>
<td>18,726</td>
</tr>
<tr>
<td>1309019</td>
<td>JT152EAA100302177</td>
<td>Silver</td>
<td>Tacoma13</td>
<td>2013</td>
<td>Car/4DR</td>
<td>20,352</td>
</tr>
<tr>
<td>1309020</td>
<td>JT152EAA100302178</td>
<td>Blue</td>
<td>Tundra13</td>
<td>2013</td>
<td>Car/4DR</td>
<td>30,107</td>
</tr>
<tr>
<td>1309021</td>
<td>JT152EAA100302179</td>
<td>Blue</td>
<td>Prius13</td>
<td>2013</td>
<td>Car/4DR</td>
<td>29,214</td>
</tr>
<tr>
<td>1309022</td>
<td>JT152EAA100302180</td>
<td>Green</td>
<td>Prius13</td>
<td>2013</td>
<td>Car/4DR</td>
<td>19,273</td>
</tr>
<tr>
<td>1309023</td>
<td>JT152EAA100302181</td>
<td>Orange</td>
<td>Prius13</td>
<td>2013</td>
<td>Car/4DR</td>
<td>19,273</td>
</tr>
</tbody>
</table>
### EMPLOYEE

<table>
<thead>
<tr>
<th>Employee_ID</th>
<th>FirstName</th>
<th>LastName</th>
<th>Position</th>
<th>EAddress</th>
<th>Ecity</th>
<th>Estate</th>
<th>EZip</th>
<th>CPhone</th>
<th>HPhone</th>
</tr>
</thead>
<tbody>
<tr>
<td>101</td>
<td>John</td>
<td>Gonzalez</td>
<td>Salesman</td>
<td>1405 Candace Ave</td>
<td>Bakersfield</td>
<td>CA</td>
<td>93307</td>
<td>6617568914</td>
<td>6613969687</td>
</tr>
<tr>
<td>102</td>
<td>Dwayne</td>
<td>McGee</td>
<td>Salesman</td>
<td>1809 Sunset Dr.</td>
<td>Bakersfield</td>
<td>CA</td>
<td>93308</td>
<td>6615697894</td>
<td>6615236987</td>
</tr>
<tr>
<td>103</td>
<td>Joe</td>
<td>Biden</td>
<td>Mechanic</td>
<td>143 20TH ST.</td>
<td>Bakersfield</td>
<td>CA</td>
<td>93311</td>
<td>6615235476</td>
<td>6614563210</td>
</tr>
<tr>
<td>104</td>
<td>Maggie</td>
<td>Flores</td>
<td>Salesman</td>
<td>456 King St.</td>
<td>Shafter</td>
<td>CA</td>
<td>93263</td>
<td>6619848452</td>
<td>6612368547</td>
</tr>
<tr>
<td>105</td>
<td>Zenson</td>
<td>Ponce</td>
<td>Mechanic</td>
<td>4857 Bell Ave.</td>
<td>Bakersfield</td>
<td>CA</td>
<td>93309</td>
<td>6619856985</td>
<td>6615201456</td>
</tr>
<tr>
<td>106</td>
<td>Raymond</td>
<td>Dunn</td>
<td>Salesman</td>
<td>890 Gilmore Way</td>
<td>Taft</td>
<td>CA</td>
<td>93268</td>
<td>6612153258</td>
<td>6617459620</td>
</tr>
<tr>
<td>107</td>
<td>George</td>
<td>Vazquez</td>
<td>Mechanic</td>
<td>6789 Spruce st.</td>
<td>Bakersfield</td>
<td>CA</td>
<td>93310</td>
<td>6612547546</td>
<td>6611248541</td>
</tr>
</tbody>
</table>

### SERVICETYPE

<table>
<thead>
<tr>
<th>Service_id</th>
<th>ServiceName</th>
<th>Service_Date</th>
<th>Result</th>
<th>Vehicle_id</th>
<th>Employee_id</th>
</tr>
</thead>
<tbody>
<tr>
<td>13090010101</td>
<td>Brake Inspection</td>
<td>10/2/2013</td>
<td>Brake Recall Repaired</td>
<td>1309001</td>
<td>103</td>
</tr>
<tr>
<td>13090010201</td>
<td>Brake Inspection</td>
<td>10/2/2013</td>
<td>Brake Recall Repaired</td>
<td>1309002</td>
<td>107</td>
</tr>
<tr>
<td>13090010301</td>
<td>General Inspection</td>
<td>10/3/2013</td>
<td>All systems good</td>
<td>1309003</td>
<td>103</td>
</tr>
<tr>
<td>13090010401</td>
<td>Brake Inspection</td>
<td>10/3/2013</td>
<td>Brake Recall Repaired</td>
<td>1309004</td>
<td>107</td>
</tr>
<tr>
<td>13090010501</td>
<td>General Inspection</td>
<td>10/3/2013</td>
<td>All systems good</td>
<td>1309005</td>
<td>103</td>
</tr>
<tr>
<td>13090010601</td>
<td>Transmission Inspection</td>
<td>10/3/2013</td>
<td>Clutch Recall repaired</td>
<td>1309006</td>
<td>107</td>
</tr>
<tr>
<td>13090010701</td>
<td>General Inspection</td>
<td>10/4/2013</td>
<td>All systems good</td>
<td>1309007</td>
<td>103</td>
</tr>
<tr>
<td>13090010801</td>
<td>Transmission Inspection</td>
<td>10/4/2013</td>
<td>Clutch recall repaired</td>
<td>1309008</td>
<td>107</td>
</tr>
<tr>
<td>13090010901</td>
<td>General Inspection</td>
<td>10/4/2013</td>
<td>All systems good</td>
<td>1309009</td>
<td>103</td>
</tr>
<tr>
<td>13090011001</td>
<td>Transmission Inspection</td>
<td>10/4/2013</td>
<td>All systems good</td>
<td>1309010</td>
<td>107</td>
</tr>
<tr>
<td>13090011101</td>
<td>Brake Inspection</td>
<td>10/5/2013</td>
<td>Brake Recall Repaired</td>
<td>1309011</td>
<td>103</td>
</tr>
<tr>
<td>13090011201</td>
<td>Brake Inspection</td>
<td>10/5/2013</td>
<td>All systems good</td>
<td>1309012</td>
<td>107</td>
</tr>
<tr>
<td>13090011301</td>
<td>Engine Inspection</td>
<td>10/5/2013</td>
<td>All systems good</td>
<td>1309013</td>
<td>103</td>
</tr>
<tr>
<td>13090011401</td>
<td>Transmission Inspection</td>
<td>10/5/2013</td>
<td>All systems good</td>
<td>1309014</td>
<td>107</td>
</tr>
<tr>
<td>13090011501</td>
<td>Engine Inspection</td>
<td>10/5/2013</td>
<td>All systems good</td>
<td>1309015</td>
<td>103</td>
</tr>
<tr>
<td>13090011601</td>
<td>Transmission Inspection</td>
<td>10/5/2013</td>
<td>All systems good</td>
<td>1309016</td>
<td>107</td>
</tr>
<tr>
<td>13090011701</td>
<td>Engine Inspection</td>
<td>10/24/2013</td>
<td>All systems good</td>
<td>1309017</td>
<td>103</td>
</tr>
<tr>
<td>13090011801</td>
<td>Transmission Inspection</td>
<td>10/24/2013</td>
<td>All systems good</td>
<td>1309018</td>
<td>107</td>
</tr>
<tr>
<td>13090011901</td>
<td>Brake Inspection</td>
<td>10/24/2013</td>
<td>All systems good</td>
<td>1309019</td>
<td>103</td>
</tr>
<tr>
<td>13090012001</td>
<td>Engine Inspection</td>
<td>10/25/2013</td>
<td>All systems good</td>
<td>1309020</td>
<td>107</td>
</tr>
<tr>
<td>13090012101</td>
<td>Transmission Inspection</td>
<td>10/25/2013</td>
<td>Clutch Recall repaired</td>
<td>1309021</td>
<td>103</td>
</tr>
<tr>
<td>13090012201</td>
<td>General Inspection</td>
<td>10/25/2013</td>
<td>All systems good</td>
<td>1309022</td>
<td>107</td>
</tr>
<tr>
<td>13090012301</td>
<td>Engine Inspection</td>
<td>10/25/2013</td>
<td>All systems good</td>
<td>1309023</td>
<td>103</td>
</tr>
</tbody>
</table>
### CUSTOMER(Customer_id, CFirstName, CLastName, CAddress, CCity, CState, CZip, Phone)

<table>
<thead>
<tr>
<th>Customer_id</th>
<th>CFirstName</th>
<th>CLastName</th>
<th>CAddress</th>
<th>CCity</th>
<th>CState</th>
<th>CZip</th>
</tr>
</thead>
<tbody>
<tr>
<td>10001</td>
<td>Mary</td>
<td>Valastro</td>
<td>7896 14th St.</td>
<td>Bakersfield</td>
<td>CA</td>
<td>93313</td>
</tr>
<tr>
<td>10002</td>
<td>Guadalupe</td>
<td>Gonzalez</td>
<td>532 Larson St.</td>
<td>Bakersfield</td>
<td>CA</td>
<td>93311</td>
</tr>
<tr>
<td>10003</td>
<td>Joseph</td>
<td>Villasenor</td>
<td>2485 Village Dr.</td>
<td>Bakersfield</td>
<td>CA</td>
<td>93311</td>
</tr>
<tr>
<td>10004</td>
<td>Nataly</td>
<td>Enciso</td>
<td>4851 Mills Ave.</td>
<td>Shafter</td>
<td>CA</td>
<td>93263</td>
</tr>
<tr>
<td>10005</td>
<td>Nikki</td>
<td>Procell</td>
<td>3269 Carnage St.</td>
<td>Shafter</td>
<td>CA</td>
<td>93263</td>
</tr>
<tr>
<td>10006</td>
<td>Baleigh</td>
<td>Hosey</td>
<td>4532 Cardona Dr.</td>
<td>Bakersfield</td>
<td>CA</td>
<td>93307</td>
</tr>
<tr>
<td>10007</td>
<td>Marissa</td>
<td>Delatorre</td>
<td>9756 Nixon Dr.</td>
<td>Fresno</td>
<td>CA</td>
<td>93611</td>
</tr>
<tr>
<td>10008</td>
<td>Johnnie</td>
<td>Watts</td>
<td>966 Lory Vill</td>
<td>Taft</td>
<td>CA</td>
<td>93268</td>
</tr>
<tr>
<td>10009</td>
<td>Chris</td>
<td>Darell</td>
<td>1695 Finch Way</td>
<td>Bakersfield</td>
<td>CA</td>
<td>93307</td>
</tr>
<tr>
<td>10010</td>
<td>Reuben</td>
<td>Reyes</td>
<td>7989 Sissy St.</td>
<td>Fresno</td>
<td>CA</td>
<td>93611</td>
</tr>
<tr>
<td>10011</td>
<td>Karie</td>
<td>Johnson</td>
<td>6546 Miles Dr.</td>
<td>Bakersfield</td>
<td>CA</td>
<td>93312</td>
</tr>
<tr>
<td>10012</td>
<td>Nicki</td>
<td>Jackson</td>
<td>2463 Hughes Ave.</td>
<td>Bakersfield</td>
<td>CA</td>
<td>93308</td>
</tr>
<tr>
<td>10013</td>
<td>Noemi</td>
<td>Gonzalez</td>
<td>6713 Peg Way</td>
<td>Bakersfield</td>
<td>CA</td>
<td>93309</td>
</tr>
<tr>
<td>10014</td>
<td>Gabriel</td>
<td>Zune</td>
<td>8008 Stockdale.</td>
<td>Bakersfield</td>
<td>CA</td>
<td>93311</td>
</tr>
<tr>
<td>10015</td>
<td>Gladis</td>
<td>Munoz</td>
<td>2598 Gisbo ST.</td>
<td>Bakersfield</td>
<td>CA</td>
<td>93308</td>
</tr>
</tbody>
</table>

## 5. SAMPLE QUERIES

In this section, we will create a list of queries which will return tuples from this database that match the selection. The resultant queries will be presented in English and in three relational languages used to represent these queries. The three languages are: relational algebra, relational calculus, and domain relational calculus.

### 5.1 Design of Queries

MANUFACTURER (Manufacturer_id, Name, Address, City, St, Zip, Phone)

VEHICLETYPE(Model_id, Year, Model, Type, MSRP)
VEHICLES (Vehicles_id, VIN, Miles, Color, Model_id, Aq_Date, Aq_Price, Manufacturer_id, Service_id, Sale_Date, Sale_Price, Commission, Employee_id, Customer_id, )

EMPLOYEE(Employee_id, FirstName, LastName, Position, EAddress, EState, ECity, EZip, CPhone, HPhone)

CUSTOMER(Customer_id, CFirstName, CLastName, CAddress, CCity, CState, CZip, Phone)

SERVICETYPE(Service_id, Service_Name, Service_date, Result, Vehicle_id, Employee_id)

1. List all Vehicles that John Gonzalez has Sold.
2. List all Vehicles that had a ‘General Inspection’.
3. List the Acquired Price of Vehicle 1309004.
4. List all Customers who have Purchased a Camry.
5. List all Customers who were sold a car by Dwayne McGee.
7. List all Vehicles not Sold by Maggie Flores.
8. Select all Vehicles sold on 10/15/13.
9. List vehicles that cost over $25,000.
10. List Vehicles that cost under $25,000.

5.2 Relational Algebra, Tuple Relational Calculus and Domain Relational Calculus of Queries

Relational Algebra

- Relational Algebra consists of ordered material and complete sets of algebraic functions, the primary function of relational algebra is giving a groundwork set up for the query languages in relational databases.

Tuple Relational Calculus
- As introduced by Edgar Codd as being a strong part of the relational models, is a type of calculus to show the deposition database query language for relational models.

**Domain Relational Calculus**

- Domain Relational Calculus is another calculus used for database query language for relational models, also DRC have their own distinct form including the variables, constants, and quantifiers.

1. **List all Vehicles that John Gonzalez has Sold.**

   **Relational Algebra:**
   
   \[ \Pi_{v.model\_id, v.vehicle\_id} (\sigma_{v.employee\_id = e.employee\_id, e.firstname = 'John' \land e.lastname = 'Gonzalez'} Vehicle^y \times Employee^e ) \]

   **Tuple Relational Calculus:**
   
   \{ v.model\_id, v.vehicle\_id | Vehicles(v) \land (\exists e) Employee(e) \land v.employee\_id = e.employee\_id \land e.firstname = 'John' \land e.lastname = 'Gonzalez'\}\}

   **Domain Relational Calculus:**
   
   \{ v,m | (\exists v)(Vehicles(v, m, _ , _ , _ , _ , _ , _ , _ , _ , _ , _ , _ )) \land (\exists e)(Employee(e, 'John', 'Gonzalez', _ , _ , _ , _ , _ , _ ))\}\}

2. **List all Vehicles that had a ‘General Inspection’**.

   **Relational Algebra:**
   
   \[ \Pi_{vehicle\_id,*} (\sigma_{s.servicename = 'General Inspection'} (SERVICETYPE^s)) \]

   **Tuple Relational Calculus:**
   
   \{ s.vehicle\_id | SERVICETYPE(s) \land s.servicename = 'General Inspection'\}\}
Domain Relational Calculus:

\{ v \mid (\exists v) \, (\text{SERVICETYPE}(\_ , \text{‘General Inspection, } \_,\_\_, \_)) \}

3. **List the Acquired Price of Vehicle 1309004.**

Relational Algebra:

\[ \Pi_{v.aq\_price} (\sigma_{v.vehicle\_id = '1309004'}(\text{Vehicles})) \]

Tuple Relational Calculus:

\{ v.aq\_price \mid \text{Vehicles}(v) \land v.vehicle\_id = 1309004 \}

Domain Relational Calculus:

\{ v, a \mid (\exists v)(\text{Vehicles}(v,\_\_\_\_\_\_\_\_, a,\_\_\_\_\_\_\_\_)) \}

4. **List all Customers who have purchased a Camry.**

Relational Algebra:

\[ \Pi_{c1.firstname, c1.lastname}(\text{Customer}^{c1} \star (\Pi_{c\_customer\_id} \text{Customer}^{c2} - \Pi_{v.vehicle\_id} (\sigma_{v.model\_id = t.model\_id} \land t.model\_id \neq 'camry13' \times \text{VehicleType}^{t} \times \text{Vehicles}^{v})) \]

Tuple Relational Calculus:

\{ c \mid \text{Customer}(c) \land (\neg (\exists c2)(\text{Customer}(c2) \land (\exists t)(\text{VehicleType}(t) \land (\exists v)(\text{Vehicles}(v) \land \\
\quad c2.customer\_id = v.customer\_id \land v.model\_id = t.model\_id \land t.model\_id \neq ‘Camry13’)))}) \}

Domain Relational Calculus:

\{ c, l \mid \text{Customer}(i, c, l,\_\_\_\_\_\_) \land (\forall v)(\text{Vehicles}(\_\_\_\_\_\_\_, \_\_\_\_\_\_, c, t) \land (\exists t)(\text{VehicleType}(‘Camry13’, \_\_\_\_\_, \_\_\_\_\_\_\_))} \}

5. **List all Customers who were sold a car by Dwayne McGee.**

Relational Algebra

\[ \Pi_{v.customer\_id} (\sigma_{v.employee\_id = e.employee\_id \land e.firstname = ‘Dwayne’ \land e.lastname = ‘McGee’} (\text{Vehicles}^{v} \times \text{Employee}^{e}))) \]
Tuple Relational Calculus:

{ c2 | (Customer(c2) ^ (Ǝv) (Vehicles(v) ^ (Ǝe)(Employee(e) ^ c2.customer_id = v.customer_id ^ v.employee_id = e.employee_id ^ e.efirstname = ’Dwayne’ ^ e.elastname = ’McGee’))))}

Domain Relational Calculus:

\{ c | (Ǝc)(Customer(i, c, _, _, _, _) ^ (Ǝv)(Vehicles(_, _, _, _, _, i, e, _) ^ (Ǝe) (Employee(e, ‘Dwayne’, ‘McGee’, _, _, _, _, _, _))))}\}


Relational Algebra:

\[ \Pi_{s.vehicle\_id} (\sigma_{s.service\_date = '10/5/2013'} ^ s.employee\_id = e.employee\_id ^ e.efirstname = 'Joe' ^ e.elastname = 'Biden' (Employee e)) \times ServicedType s \]

Tuple Relational Calculus:

{ s.vehicle_id | ServiceType (s) ^ (Ǝe)(Employee(e) ^ e.employee_id = s.employee_id ^ e.efirstname = ‘Joe’ ^ e.elastname = ‘Biden’ ^ s.service_date = ‘5-OCT-2013’)}

Domain Relational Calculus:

\{v | (Ǝs)( ServiceType(_,_,,’10/5/2013’,_,e,v) ^ (Ǝe)(Employee(e, ‘Joe’, ‘Biden’, _,_,_,_,_,_,_))))\}

7. List all Vehicles not Sold by Maggie Flores.

Relational Algebra:

\[ \Pi_{V.*} (\sigma_{V.employee\_id = e.employee\_id ^ e.efirstname ≠ 'Maggie' ^ e.elastname ≠ 'Flores'} (Vehicles V \times Employee e)) \]

Tuple Relational Calculus:

{ v | Vehicles(v) ^ (Ǝe)(Employee(e)^ v.employee_id=e.employee_id ^ e.firstname ≠ ’Maggie’ ^ e.lastname ≠ ’Flores’)}

Domain Relational Calculus:

\{v[Vehicles(v, _,_,_,_,e,_) ^- (Ǝe)(Employee(e, ‘Maggie’, ‘Flores’, _,_,_,_,_,_))}\}

8. Select all Vehicles sold on 10/15/13.

Relational Algebra
\[ \Pi_{v.*}(\sigma_{v.sale_date = '10/15/2013'}(\text{Vehicles}^v)) \]

Tuple Calculus

\{ v | \text{Vehicles}(v) \land v.sale_date = '10/15/13' \}

Domain Relational Calculus

\{ v | \text{Vehicles}(v, ____________, ____________, '10/15/13', ____________) \}

9. List vehicles that cost over $25,000.

Relational Algebra:

\[ \pi_{v.*}( \sigma_{v1.vehicle_id \neq v2.vehicle_id \land v1.aq_price > 25000} (\text{Vehicles}^{v1} \times \text{Vehicles}^{v2}) ) \]

Tuple Calculus

\{ v | (\exists v) \text{Vehicles}(v) \land v.aq_price > 25000 \}

Domain Relational Calculus

\{ v | (\exists v) \text{Vehicles}(v, ____________, ____________, c > '25000', ____________) \}

10. List vehicles that cost under $25,000.

Relational Algebra

\[ \Pi_{v.*}(\sigma_{v1.vehicle_id \neq v2.vehicle_id \land v2.aq_price != v1.aq_price \land v2.price < 25000} (\text{Vehicles}^{v1} \times \text{Vehicles}^{v2})) \]

Tuple Calculus

\{ v | (\exists v) \text{Vehicles}(v) \land v.aq_price < 25000 \}

Domain Relational Calculus

\{ v | (\exists v) \text{Vehicle}(v, ____________, c < 25000, ____________) \}
Phase III

6. NORMALIZATION OF RELATIONS

6.1.1. Data Normalization

Normalization is the set of rules and techniques used to identify relationships among attributes, combine attributes to form relations, and combining relations to form a database. The goal of data normalization is to eliminate redundant data storage, construct data so that model is flexible, and close modeling of real world entities, processes, and their relationships. It is the best way to efficiently organize data in a database.

6.1.2 Definitions of Normal Forms:

• **First Normal Form:** is where you have a table in the first normal form only if all the key attributes are defined and repetitions of groups are available.

• **Second Normal Form:** is configured only if a table is in its first normal form along with no main attributes with full functionality as a dependent, along with no partial dependencies and is on the entire primary key.

• **Third Normal Form:** a table can only make it to the third normal form only if it’s in the second normal form and every non-key attribute has no transitive dependencies on the primary key.

• **Boyce-Codd Normal Form:** can only be considered IF AND ONLY IF a table has made it in the third normal for along with every determinant is the candidate key.

6.1.3 Relations Normal Form Check

Employee: First Normal Form (2NF)
VehicleType: Second Normal Form (2NF)
Manufacturer: First Normal Form (1NF)
Vehicles: First Normal Form (1NF)
ServiceType: Second Normal Form (2NF)
7. SQL *PLUS

SQL *Plus is one of the features installed with oracle 11g dbms package. It is a basic command-line interface, most commonly used by user, administrator, and programmers. SQL *Plus has its own commands, which provides access to the Oracle Database. This tool can be used to create, manage, and query data in the oracle database. With the help of SQL *Plus we will execute the queries from phase II and any other queries.

8. SCHEMA OBJECTS IN ORACLE

A schema is a gathering of logical structures of data, which is put together with schema objects. These are then stored logically within a tablespace of the database by Oracle databases. The schema objects can be created and manipulated using SQL commands. Below you will see the schema objects used for this database management system.

8.1 Tables

Tables are the bases of data storage in Oracle database and store the data of entities or relations. Tables are defined with a table name and column sets (like vehicles). Each column has a column name (like vehicle_id and model), a datatype (like varchar or int), and integrity constraints (like not null). In SQL *Plus the create tables will follow the following format allowing also permissions to set primary keys, foreign keys, and unique structures.

CREATE TABLE table_name
(  
column1 datatype null/not null,  
column2 datatype null/not null,  
....
);

8.2 Views

A view is a representation of the data in one or more tables. The view shows the output of a query as a table, which thought as a virtual table. The neat thing about view is that you may controlled to only show specified columns in a table, in order to present data to different types of users.

CREATE VIEW view_name AS
SELECT columns
FROM table
WHERE predicates;

8.3 Sequences

A sequence or sequence generator provides a constant flow of numbers which is used in many cases where the assigned number is unique. Sequences in a relational database are normally used to create ID numbers for specified relations who are then used as primary keys.

CREATE SEQUENCE sequence_name
MINVALUE value
MAXVALUE value
START WITH value
INCREMENT BY value
CACHE value;

8.4 Synonyms

Synonyms are aliases for tables, views, materialized views, sequences, procedures, functions, or packages. The purpose for using synonyms is to secure schema objects, or make them more convenient to other database users.

CREATE [OR REPLACE] [PUBLIC] synonym [schema.] synonym_name
For [schema.] object_name [@dblink];
8.5 Indexes

Indexes are structures that were supplemented with tables and clusters. Indexes can be created on many columns of a table to speed up SQL statement execution on specified tables. Creating an index will provide a faster access to table data.

CREATE [UNIQUE] INDEX index_name
    ON table_name (column1, column2,... column_n)
    [COMPUTE STATISTICS];

8.6 Clusters

Clusters are another way of storing table data. A cluster is a combination of two or more tables that has the same data blocks because some tables share the same columns. Creating clusters helps run queries much faster.

CREATE CLUSTER cluster_name (column1, column2,... column_n)
    SIZE dataBlockSize
    STORAGE (INITIAL initialSizeOfCluster NEXT sizeToAllocate);

8.7 Database Links

Database links is when one database connects to another one in means of accessing and querying data in that other database. However the link only works one way, so the database being accessed cannot access the first database.

CREATE DATABASE LINK link_name
    CONNECT TO CURRENT_USER
    USING connect_string;

8.8 Snapshots

A snapshot is the state of data in a table at a certain point in time. It helps to understand how the data changed from one point in time to its current state.

CREATE MATERIALIZED VIEW schema_name.snapsho_name
    AS select_statement;
8.9 Procedures

Commonly known as stored procedures is a subroutine that summarizes the steps to access data in a database. The procedures can be written in a variety of programming languages, but it all depends on the database system. The most commonly used is PL/SQL.

CREATE [OR REPLACE] PROCEDURE procedure_name
    ((parameter [.parameter]))
IS
    [declaration_section]
BEGIN
    executable_section
[EXCEPTION
    Exception_section]
END [procedure_name];

8.10 Function

As you can see, the function follows a similar format to that of Procedures. The difference is that functions return a single value or variable, while procedures return multiple values.

CREATE [OR REPLACE] FUNCTION function_name
    ((parameter [.parameter]))
    RETURN return_datatype
IS | AS
    [declaration_section]
BEGIN
    executable_section
[EXCEPTION
    exception_section]
END [function_name];

8.11 Packages

Packages are the assemblage of many related functions and procedures, which for a large set of instructions to follow throughout the execution.
9. RELATION SCHEMA AND CONTENTS

9.1 DESC JGDM_EMPLOYEE;

<table>
<thead>
<tr>
<th>Name</th>
<th>Null?</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>EMPLOYEE_ID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
</tr>
<tr>
<td>EFIRSTNAME</td>
<td></td>
<td>VARCHAR2(20)</td>
</tr>
<tr>
<td>ELASTNAME</td>
<td></td>
<td>VARCHAR2(20)</td>
</tr>
<tr>
<td>EADDRESS</td>
<td></td>
<td>VARCHAR2(40)</td>
</tr>
<tr>
<td>ECITY</td>
<td></td>
<td>VARCHAR2(40)</td>
</tr>
<tr>
<td>ESTATE</td>
<td></td>
<td>VARCHAR2(2)</td>
</tr>
<tr>
<td>EZIP</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
</tr>
<tr>
<td>HPHONE</td>
<td></td>
<td>NUMBER(38)</td>
</tr>
<tr>
<td>CPHONE</td>
<td></td>
<td>NUMBER(38)</td>
</tr>
<tr>
<td>POSITION</td>
<td>NOT NULL</td>
<td>VARCHAR2(20)</td>
</tr>
</tbody>
</table>

SELECT * FROM JGDM_EMPLOYEE;

<table>
<thead>
<tr>
<th>EMPLOYEE_ID</th>
<th>EFIRSTNAME</th>
<th>ELASTNAME</th>
<th>EADDRESS</th>
<th>ECITY</th>
<th>ES</th>
<th>EZIP</th>
<th>HPHONE</th>
<th>CPHONE</th>
<th>POSITION</th>
</tr>
</thead>
<tbody>
<tr>
<td>101</td>
<td>John</td>
<td>Gonzalez</td>
<td>1405 Candace Ave</td>
<td>Bakersfield</td>
<td>CA</td>
<td>93307</td>
<td>6617568914</td>
<td>6613969687</td>
<td>salesman</td>
</tr>
<tr>
<td>102</td>
<td>Dwayne</td>
<td>McGee</td>
<td>1809 Sunset Dr.</td>
<td>Bakersfield</td>
<td>CA</td>
<td>93308</td>
<td>6615697894</td>
<td>6615236987</td>
<td>salesman</td>
</tr>
<tr>
<td>103</td>
<td>Joe</td>
<td>Biden</td>
<td>143 20TH ST.</td>
<td>Bakersfield</td>
<td>CA</td>
<td>93311</td>
<td>6615235476</td>
<td>6614563210</td>
<td>mechanic</td>
</tr>
<tr>
<td>104</td>
<td>Maggie</td>
<td>Flores</td>
<td>456 King St.</td>
<td>Shafter</td>
<td>CA</td>
<td>93263</td>
<td>6619848452</td>
<td>6612368547</td>
<td>salesman</td>
</tr>
<tr>
<td>105</td>
<td>Zenson</td>
<td>Ponce</td>
<td>4857 Bell Ave.</td>
<td>Bakersfield</td>
<td>CA</td>
<td>93309</td>
<td>6619856985</td>
<td>6615201456</td>
<td>salesman</td>
</tr>
<tr>
<td>106</td>
<td>Raymond</td>
<td>Dunn</td>
<td>890 Gilmore Way</td>
<td>Taft</td>
<td>CA</td>
<td>93268</td>
<td>6612153258</td>
<td>6617459620</td>
<td>salesman</td>
</tr>
<tr>
<td>107</td>
<td>George</td>
<td>Vazquez</td>
<td>6789 Spruce st.</td>
<td>Bakersfield</td>
<td>CA</td>
<td>93310</td>
<td>6612547546</td>
<td>6611248541</td>
<td>mechanic</td>
</tr>
</tbody>
</table>
9.2 DESC JGDM_VEHICLES

<table>
<thead>
<tr>
<th>Name</th>
<th>Null?</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>VEHICLE_ID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
</tr>
<tr>
<td>MODEL_ID</td>
<td>NOT NULL</td>
<td>VARCHAR2(20)</td>
</tr>
<tr>
<td>COLOR</td>
<td>NOT NULL</td>
<td>VARCHAR2(20)</td>
</tr>
<tr>
<td>VIN</td>
<td>NOT NULL</td>
<td>VARCHAR2(40)</td>
</tr>
<tr>
<td>AQ_DATE</td>
<td>NOT NULL</td>
<td>DATE</td>
</tr>
<tr>
<td>AQ_PRICE</td>
<td>NOT NULL</td>
<td>FLOAT(126)</td>
</tr>
<tr>
<td>MILES</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
</tr>
<tr>
<td>CUSTOMER_ID</td>
<td></td>
<td>NUMBER(38)</td>
</tr>
<tr>
<td>EMPLOYEE_ID</td>
<td></td>
<td>NUMBER(38)</td>
</tr>
<tr>
<td>MANUFACTURER_ID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
</tr>
<tr>
<td>SALE_DATE</td>
<td></td>
<td>DATE</td>
</tr>
<tr>
<td>SALE_PRICE</td>
<td></td>
<td>FLOAT(126)</td>
</tr>
<tr>
<td>COMMISSION</td>
<td></td>
<td>FLOAT(126)</td>
</tr>
<tr>
<td>SERVICE_ID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
</tr>
</tbody>
</table>

SELECT * FROM JGDM_VEHICLES

<table>
<thead>
<tr>
<th>VEHICLE_ID</th>
<th>MODEL_ID</th>
<th>COLOR</th>
<th>VIN</th>
<th>AQ_DATE</th>
<th>AQ_PRICE</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>MILES</th>
<th>CUSTOMER_ID</th>
<th>EMPLOYEE_ID</th>
<th>MANUFACTURER_ID</th>
<th>SALE_DATE</th>
<th>SALE_PRICE</th>
<th>COMMISSION</th>
<th>SERVICE_ID</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>10654.93</td>
<td>130900101</td>
<td>10011</td>
<td>101</td>
<td>21831</td>
<td>654.93</td>
<td></td>
<td>130900101</td>
</tr>
</tbody>
</table>

7 rows selected.
<table>
<thead>
<tr>
<th>VIN</th>
<th>Make</th>
<th>Model</th>
<th>Color</th>
<th>License Plate</th>
<th>Date</th>
<th>Mileage</th>
<th>Reg No</th>
<th>Status</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>1309002</td>
<td>Camry</td>
<td>13</td>
<td>BLACK</td>
<td>JT152EEA100302160</td>
<td>01-Sep-13</td>
<td>21531</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1309003</td>
<td>Camry</td>
<td>13</td>
<td>BLACK</td>
<td>JT152EEA100302161</td>
<td>02-Sep-13</td>
<td>21531</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1309004</td>
<td>Camry</td>
<td>13</td>
<td>WHITE</td>
<td>JT152EEA100302162</td>
<td>02-Sep-13</td>
<td>22497</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1309005</td>
<td>CamryHybrid</td>
<td>13</td>
<td>BLACK</td>
<td>JT152EEA100302163</td>
<td>02-Sep-13</td>
<td>25302</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1309006</td>
<td>CamryHybrid</td>
<td>13</td>
<td>WHITE</td>
<td>JT152EEA100302164</td>
<td>02-Sep-13</td>
<td>25302</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1309007</td>
<td>CamryHybrid</td>
<td>13</td>
<td>GREEN</td>
<td>JT152EEA100302165</td>
<td>03-Sep-13</td>
<td>25302</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1309008</td>
<td>Avalon</td>
<td>13</td>
<td>GREEN</td>
<td>JT152EEA100302166</td>
<td>03-Sep-13</td>
<td>29214</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1309009</td>
<td>Avalon</td>
<td>13</td>
<td>RED</td>
<td>JT152EEA100302167</td>
<td>03-Sep-13</td>
<td>29214</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1309010</td>
<td>AvalonHybrid</td>
<td>13</td>
<td>Silver</td>
<td>JT152EEA100302168</td>
<td>04-Sep-13</td>
<td>33396</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1309011</td>
<td>AvalonHybrid</td>
<td>13</td>
<td>White</td>
<td>JT152EEA100302169</td>
<td>04-Sep-13</td>
<td>33396</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1309012</td>
<td>Highlander</td>
<td>13</td>
<td>Blue</td>
<td>JT152EEA100302170</td>
<td>04-Sep-13</td>
<td>27568</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VEHICLE_ID</td>
<td>MODEL_ID</td>
<td>COLOR</td>
<td>VIN</td>
<td>AQ_DATE</td>
<td>AQ_PRICE</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>------------</td>
<td>----------</td>
<td>-------</td>
<td>-------------------</td>
<td>----------</td>
<td>----------</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1309013</td>
<td>Highlander 13</td>
<td>White</td>
<td>JT152EEA100302171</td>
<td>04-SEP-13</td>
<td>27568</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>10004</td>
<td>103</td>
<td>4-21-OCT-13</td>
<td>22697</td>
<td>680.91</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1309014</td>
<td>Highlander 13</td>
<td>Red</td>
<td>JT152EEA100302172</td>
<td>04-SEP-13</td>
<td>27568</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>10005</td>
<td>104</td>
<td>4-24-OCT-13</td>
<td>28688</td>
<td>860.64</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1309015</td>
<td>FJ Cruiser 13</td>
<td>Silver</td>
<td>JT152EEA100302173</td>
<td>04-SEP-13</td>
<td>26707</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>10006</td>
<td>105</td>
<td>6-25-OCT-13</td>
<td>25502</td>
<td>765.06</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1309016</td>
<td>FJ Cruiser 213</td>
<td>Black</td>
<td>JT152EEA100302174</td>
<td>25-SEP-13</td>
<td>27577</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>10007</td>
<td>106</td>
<td>6-26-OCT-13</td>
<td>28400</td>
<td>852.00</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1309017</td>
<td>Sienna 13</td>
<td>Gold</td>
<td>JT152EEA100302175</td>
<td>25-SEP-13</td>
<td>25905</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>10008</td>
<td>101</td>
<td>3-26-OCT-13</td>
<td>29514</td>
<td>885.42</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1309018</td>
<td>Tacoma 13</td>
<td>White</td>
<td>JT152EEA100302176</td>
<td>25-SEP-13</td>
<td>18726</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>10009</td>
<td>102</td>
<td>3-26-OCT-13</td>
<td>30100</td>
<td>903.00</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1309019</td>
<td>Tacoma 13</td>
<td>Silver</td>
<td>JT152EEA100302177</td>
<td>25-SEP-13</td>
<td>20352</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>10010</td>
<td>104</td>
<td>4-27-OCT-13</td>
<td>34000</td>
<td>1020.00</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1309020</td>
<td>Tundra 13</td>
<td>Blue</td>
<td>JT152EEA100302178</td>
<td>25-SEP-13</td>
<td>30107</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>10013</td>
<td>105</td>
<td>1-27-OCT-13</td>
<td>24977</td>
<td>749.31</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1309021</td>
<td>Prius 13</td>
<td>Blue</td>
<td>JT152EEA100302179</td>
<td>25-SEP-13</td>
<td>23977</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>10014</td>
<td>106</td>
<td>1-27-OCT-13</td>
<td>19873</td>
<td>596.19</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
1309022 Prius13 Green JT152EEA100302180 27-SEP-13 19273
0 2 130902201

1309023 Prius13 Orange JT152EEA100302181 27-SEP-13 19273
0 2 130902301

23 rows selected.

9.3 DESC JGDM_VEHICLE_TYPE:

<table>
<thead>
<tr>
<th>Name</th>
<th>Null?</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>MODEL_ID</td>
<td>NOT NULL</td>
<td>VARCHAR2(20)</td>
</tr>
<tr>
<td>YEAR</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
</tr>
<tr>
<td>TYPES</td>
<td>NOT NULL</td>
<td>VARCHAR2(10)</td>
</tr>
<tr>
<td>MSRP</td>
<td>NOT NULL</td>
<td>FLOAT(126)</td>
</tr>
</tbody>
</table>

SELECT * FROM JGDM_VEHICLE_TYPE

<table>
<thead>
<tr>
<th>MODEL_ID</th>
<th>YEAR</th>
<th>TYPES</th>
<th>MSRP</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>2013</td>
<td>CAR/4DR</td>
<td>21951</td>
</tr>
<tr>
<td>Camry13</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CamryHybrid13</td>
<td>2013</td>
<td>CAR/4DR</td>
<td>25684</td>
</tr>
<tr>
<td>Avalon13</td>
<td></td>
<td></td>
<td>29788</td>
</tr>
<tr>
<td>AvalonHybrid13</td>
<td>2013</td>
<td>CAR/4DR</td>
<td>33986</td>
</tr>
<tr>
<td>Highlander13</td>
<td>2013</td>
<td>TRUCK/4X2</td>
<td>28101</td>
</tr>
<tr>
<td>FJCruiser113</td>
<td>2013</td>
<td>TRUCK/4X2</td>
<td>27207</td>
</tr>
<tr>
<td>FJCruiser213</td>
<td>2013</td>
<td>TRUCK/4X4</td>
<td>28652</td>
</tr>
<tr>
<td>Sienna13</td>
<td></td>
<td></td>
<td>26678</td>
</tr>
<tr>
<td>Tacoma13</td>
<td></td>
<td></td>
<td>19756</td>
</tr>
<tr>
<td>Tundra13</td>
<td></td>
<td></td>
<td>31091</td>
</tr>
<tr>
<td>Prius13</td>
<td></td>
<td></td>
<td>20112</td>
</tr>
</tbody>
</table>

11 rows selected.
### 9.4 DESC JGDM_MANUFACTURER

<table>
<thead>
<tr>
<th>Name</th>
<th>Null?</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>MANUFACTURER_ID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
</tr>
<tr>
<td>NAME</td>
<td>NOT NULL</td>
<td>VARCHAR2(50)</td>
</tr>
<tr>
<td>ADDRESS</td>
<td>NOT NULL</td>
<td>VARCHAR2(50)</td>
</tr>
<tr>
<td>CITY</td>
<td>NOT NULL</td>
<td>VARCHAR2(20)</td>
</tr>
<tr>
<td>STREET</td>
<td>NOT NULL</td>
<td>CHAR(2)</td>
</tr>
<tr>
<td>ZIP</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
</tr>
<tr>
<td>PHONE</td>
<td></td>
<td>NUMBER(38)</td>
</tr>
</tbody>
</table>

#### SELECT * FROM JGDM_MANUFACTURER

<table>
<thead>
<tr>
<th>MANUFACTURER_ID</th>
<th>NAME</th>
<th>ADDRESS</th>
<th>CITY</th>
<th>ST</th>
<th>ZIP</th>
<th>PHONE</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Toyota Motor Manufacturing Mississippi, Inc.</td>
<td>1200 Magnolia Dr.</td>
<td>Blue Spring</td>
<td>MS</td>
<td>38828</td>
<td>6623173015</td>
</tr>
<tr>
<td>2</td>
<td>Toyota Motor Manufacturing Kentucky, Inc.</td>
<td>1001 Cherry Blossom Way</td>
<td>Georgetown</td>
<td>KY</td>
<td>40324</td>
<td>5028682000</td>
</tr>
<tr>
<td>3</td>
<td>Toyota Motor Manufacturing Texas, Inc.</td>
<td>1 Lone Star Pass</td>
<td>San Antonio</td>
<td>TX</td>
<td>78264</td>
<td>2102634000</td>
</tr>
<tr>
<td>4</td>
<td>Toyota Motor Manufacturing Indiana, INC</td>
<td>4000 Tulip Tree Dr.</td>
<td>Princeton</td>
<td>IN</td>
<td>47670</td>
<td>8123872000</td>
</tr>
<tr>
<td>5</td>
<td>Toyota Motor Manufacturing Alabama, INC</td>
<td>1 Cottonvalley Drive</td>
<td>Huntsville</td>
<td>AL</td>
<td>35810</td>
<td>2567465000</td>
</tr>
<tr>
<td>6</td>
<td>Toyota Motor Manufacturing West Virginia</td>
<td>1 Sugar Maple Ln</td>
<td>Buffalo</td>
<td>WV</td>
<td>25033</td>
<td>3049377000</td>
</tr>
</tbody>
</table>

6 rows selected.

### 9.5 DESC JGDM_CUSTOMER

<table>
<thead>
<tr>
<th>Name</th>
<th>Null?</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>CUSTOMER_ID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
</tr>
<tr>
<td>CNAME</td>
<td></td>
<td>VARCHAR2(20)</td>
</tr>
<tr>
<td>CADDRESS</td>
<td>NOT NULL</td>
<td>VARCHAR2(50)</td>
</tr>
<tr>
<td>CCITY</td>
<td>NOT NULL</td>
<td>VARCHAR2(20)</td>
</tr>
<tr>
<td>CSTATE</td>
<td>NOT NULL</td>
<td>VARCHAR2(20)</td>
</tr>
<tr>
<td>CZIP</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
</tr>
<tr>
<td>PHONE</td>
<td></td>
<td>NUMBER(38)</td>
</tr>
</tbody>
</table>
SELECT * FROM JGDM_CUSTOMER

<table>
<thead>
<tr>
<th>CUSTOMER_ID</th>
<th>CNAME</th>
<th>CADDRESS</th>
<th>CCITY</th>
<th>CSTATE</th>
<th>CZIP</th>
<th>PHONE</th>
</tr>
</thead>
<tbody>
<tr>
<td>10001</td>
<td>Mary Valastro</td>
<td>7896 14th St</td>
<td>Bakersfield</td>
<td>CA</td>
<td>93313</td>
<td>6616874563</td>
</tr>
<tr>
<td>10002</td>
<td>Guadalupe Gonzalez</td>
<td>532 Larson St.</td>
<td>Bakersfield</td>
<td>CA</td>
<td>93311</td>
<td>6615960987</td>
</tr>
<tr>
<td>10003</td>
<td>Joseph Villasenor</td>
<td>2485 Village Dr.</td>
<td>Bakersfield</td>
<td>CA</td>
<td>93311</td>
<td>3235673445</td>
</tr>
<tr>
<td>10004</td>
<td>Nataly Enciso</td>
<td>4851 Mills Ave.</td>
<td>Shafter</td>
<td>CA</td>
<td>93263</td>
<td>6612889877</td>
</tr>
<tr>
<td>10005</td>
<td>Nikki Procell</td>
<td>3269 Camage St.</td>
<td>Shafter</td>
<td>CA</td>
<td>93263</td>
<td>6616773455</td>
</tr>
<tr>
<td>10006</td>
<td>Baleigh Hosey</td>
<td>4532 Cardona Dr.</td>
<td>Bakersfield</td>
<td>CA</td>
<td>93307</td>
<td>6617773444</td>
</tr>
<tr>
<td>10007</td>
<td>Marissa Delatorre</td>
<td>9756 Nixon Dr.</td>
<td>Fresno</td>
<td>CA</td>
<td>93611</td>
<td>6618393232</td>
</tr>
<tr>
<td>10008</td>
<td>Johnnie Watts</td>
<td>966 Lory Vill</td>
<td>Taft</td>
<td>CA</td>
<td>93268</td>
<td>6612828277</td>
</tr>
<tr>
<td>10009</td>
<td>Chris Darell</td>
<td>1695 Finch Way</td>
<td>Bakersfield</td>
<td>CA</td>
<td>93307</td>
<td>2135787899</td>
</tr>
<tr>
<td>10010</td>
<td>Reuben Reyes</td>
<td>7989 Sissy St.</td>
<td>Fresno</td>
<td>CA</td>
<td>93611</td>
<td>6619081234</td>
</tr>
<tr>
<td>10011</td>
<td>Karie Johnson</td>
<td>6546 Miles Dr.</td>
<td>Bakersfield</td>
<td>CA</td>
<td>93312</td>
<td>6619084785</td>
</tr>
<tr>
<td>10012</td>
<td>Nicki Jackson</td>
<td>2463 Hughes Ave.</td>
<td>Bakersfield</td>
<td>CA</td>
<td>93308</td>
<td>3232434663</td>
</tr>
<tr>
<td>10013</td>
<td>Noemi Gonzalez</td>
<td>6713 Peg Way</td>
<td>Bakersfield</td>
<td>CA</td>
<td>93309</td>
<td>6611417344</td>
</tr>
<tr>
<td>10014</td>
<td>Gabriel Zune</td>
<td>8008 Stockdale.</td>
<td>Bakersfield</td>
<td>CA</td>
<td>93311</td>
<td>6614546757</td>
</tr>
<tr>
<td>10015</td>
<td>Gladis Munoz</td>
<td>2598 Gisbo ST.</td>
<td>Bakersfield</td>
<td>CA</td>
<td>93308</td>
<td>6618983433</td>
</tr>
</tbody>
</table>

15 rows selected.

9.6 DESC JGDM_SERVICETYPE

<table>
<thead>
<tr>
<th>Name</th>
<th>Null?</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>SERVICE_ID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
</tr>
<tr>
<td>SERVICENAME</td>
<td>NOT NULL</td>
<td>VARCHAR2(30)</td>
</tr>
<tr>
<td>SERVICEDATE</td>
<td>NOT NULL</td>
<td>DATE</td>
</tr>
<tr>
<td>RESULT</td>
<td>NOT NULL</td>
<td>VARCHAR2(30)</td>
</tr>
<tr>
<td>VEHICLE_ID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
</tr>
</tbody>
</table>
EMPLOYEE_ID NOT NULL NUMBER(38)

SELECT * FROM JGDM_SERVICETYPE

<table>
<thead>
<tr>
<th>SERVICE_ID</th>
<th>SERVICENAME</th>
<th>SERVICEDA</th>
<th>RESULT</th>
<th>VEHICLE_ID</th>
<th>EMPLOYEE_ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>130900101</td>
<td>Brake Inspection</td>
<td>02-OCT-13</td>
<td>Brake Recall repaired</td>
<td>1309001</td>
<td>103</td>
</tr>
<tr>
<td>130900201</td>
<td>Brake Inspection</td>
<td>02-OCT-13</td>
<td>Brake Recall repaired</td>
<td>1309002</td>
<td>107</td>
</tr>
<tr>
<td>130900301</td>
<td>General Inspection</td>
<td>03-OCT-13</td>
<td>All Systems Good</td>
<td>1309003</td>
<td>103</td>
</tr>
<tr>
<td>130900401</td>
<td>Brake Inspection</td>
<td>03-OCT-13</td>
<td>Brake Recall repaired</td>
<td>1309004</td>
<td>107</td>
</tr>
<tr>
<td>130900501</td>
<td>General Inspection</td>
<td>03-OCT-13</td>
<td>All Systems Good</td>
<td>1309005</td>
<td>103</td>
</tr>
<tr>
<td>130900601</td>
<td>Transmission Inspection</td>
<td>03-OCT-13</td>
<td>Clutch recall repaired</td>
<td>1309006</td>
<td>107</td>
</tr>
<tr>
<td>130900701</td>
<td>General Inspection</td>
<td>04-OCT-13</td>
<td>All Systems Good</td>
<td>1309007</td>
<td>103</td>
</tr>
<tr>
<td>130900801</td>
<td>Transmission Inspection</td>
<td>04-OCT-13</td>
<td>Clutch recall repaired</td>
<td>1309008</td>
<td>107</td>
</tr>
<tr>
<td>130900901</td>
<td>General Inspection</td>
<td>04-OCT-13</td>
<td>All Systems Good</td>
<td>1309009</td>
<td>103</td>
</tr>
<tr>
<td>130901001</td>
<td>Transmission Inspection</td>
<td>04-OCT-13</td>
<td>All Systems Good</td>
<td>1309010</td>
<td>107</td>
</tr>
<tr>
<td>130901101</td>
<td>Brake Inspection</td>
<td>05-OCT-13</td>
<td>Brake Recall repaired</td>
<td>1309011</td>
<td>103</td>
</tr>
<tr>
<td>130901201</td>
<td>Brake Inspection</td>
<td>05-OCT-13</td>
<td>All Systems Good</td>
<td>1309012</td>
<td>107</td>
</tr>
<tr>
<td>130901301</td>
<td>Engine Inspection</td>
<td>05-OCT-13</td>
<td>All Systems Good</td>
<td>1309013</td>
<td>103</td>
</tr>
<tr>
<td>130901401</td>
<td>Transmission Inspection</td>
<td>05-OCT-13</td>
<td>All Systems Good</td>
<td>1309014</td>
<td>107</td>
</tr>
<tr>
<td>130901501</td>
<td>Engine Inspection</td>
<td>05-OCT-13</td>
<td>All Systems Good</td>
<td>1309015</td>
<td>103</td>
</tr>
<tr>
<td>130901601</td>
<td>Transmission Inspection</td>
<td>05-OCT-13</td>
<td>All Systems Good</td>
<td>1309016</td>
<td>107</td>
</tr>
<tr>
<td>130901701</td>
<td>Engine Inspection</td>
<td>24-OCT-13</td>
<td>All Systems Good</td>
<td>1309017</td>
<td>103</td>
</tr>
<tr>
<td>130901801</td>
<td>Transmission Inspection</td>
<td>24-OCT-13</td>
<td>All Systems Good</td>
<td>1309018</td>
<td>107</td>
</tr>
<tr>
<td>130901901</td>
<td>Brake Inspection</td>
<td>24-OCT-13</td>
<td>All Systems Good</td>
<td>1309019</td>
<td>103</td>
</tr>
<tr>
<td>130902001</td>
<td>Engine Inspection</td>
<td>25-OCT-13</td>
<td>All Systems Good</td>
<td>1309020</td>
<td>107</td>
</tr>
<tr>
<td>130902101</td>
<td>Transmission Inspection</td>
<td>25-OCT-13</td>
<td>Clutch recall repaired</td>
<td>1309021</td>
<td>103</td>
</tr>
<tr>
<td>130902201</td>
<td>General Inspection</td>
<td>25-OCT-13</td>
<td>All Systems Good</td>
<td>1309022</td>
<td>107</td>
</tr>
<tr>
<td>130902301</td>
<td>Engine Inspection</td>
<td>25-OCT-13</td>
<td>All Systems Good</td>
<td>1309023</td>
<td>103</td>
</tr>
</tbody>
</table>

23 rows selected.
10. QUARIES IN SQL LANGUAGE

1. List all Vehicles that John Gonzalez has Sold.

CS342 SQL> SELECT v.model_id, v.vehicle_id FROM JGDM_VEHICLES v, JGDM_Employee e WHERE v.employee_id = e.employee_id AND e.efirstname = 'John' AND e.elastname = 'Gonzalez';

<table>
<thead>
<tr>
<th>MODEL_ID</th>
<th>VEHICLE_ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>Camry13</td>
<td>1309001</td>
</tr>
<tr>
<td>CamryHybrid13</td>
<td>1309006</td>
</tr>
<tr>
<td>Sienna13</td>
<td>1309017</td>
</tr>
</tbody>
</table>

3 rows selected

2. List all Vehicles that had a ‘General Inspection’.

CS342 SQL> SELECT s.vehicle_id FROM JGDM_SERVICETYPE s WHERE s.servicename = 'General Inspection';

<table>
<thead>
<tr>
<th>VEHICLE_ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>1309003</td>
</tr>
<tr>
<td>1309005</td>
</tr>
<tr>
<td>1309007</td>
</tr>
<tr>
<td>1309009</td>
</tr>
<tr>
<td>1309022</td>
</tr>
</tbody>
</table>

5 rows selected

3. List the Acquired Price of Vehicle 1309004.

CS342 SQL> SELECT v.aq_price FROM JGDM_VEHICLES v WHERE v.vehicle_id = '1309004';

<table>
<thead>
<tr>
<th>AQ_PRICE</th>
</tr>
</thead>
<tbody>
<tr>
<td>22497</td>
</tr>
</tbody>
</table>
4. List all Customers who have Purchased a Camry.

CS342 SQL> SELECT c2.customer_id, c2.cname FROM JGDM_CUSTOMER c2, JGDM_VEHICLES v WHERE v.model_id = 'Camry13' AND v.customer_id = c2.customer_id;

<table>
<thead>
<tr>
<th>CUSTOMER_ID</th>
<th>CNAME</th>
</tr>
</thead>
<tbody>
<tr>
<td>10001</td>
<td>Mary Valastro</td>
</tr>
<tr>
<td>10011</td>
<td>Karie Johnson</td>
</tr>
<tr>
<td>10011</td>
<td>Karie Johnson</td>
</tr>
<tr>
<td>10012</td>
<td>Nicki Jackson</td>
</tr>
</tbody>
</table>

4 rows selected

5. List all Customers who were sold a car by Dwayne McGee.

CS342 SQL> SELECT c.Customer_id, c.cname FROM JGDM_CUSTOMER c, JGDM_VEHICLES v, JGDM_EMPLOYEE e WHERE v.customer_id = c.customer_id AND v.employee_id = e.employee_id AND e.firstname = 'Dwayne' AND e.lastname = 'McGee' ORDER BY c.cname;

<table>
<thead>
<tr>
<th>CUSTOMER_ID</th>
<th>CNAME</th>
</tr>
</thead>
<tbody>
<tr>
<td>10009</td>
<td>Chris Darell</td>
</tr>
<tr>
<td>10003</td>
<td>Joseph Villasenor</td>
</tr>
<tr>
<td>10012</td>
<td>Nicki Jackson</td>
</tr>
</tbody>
</table>

3 rows selected


CS342 SQL> select v.vehicle_id, v.model_id FROM JGDM_VEHICLES v, JGDM_SERVICETYPE s, JGDM_EMPLOYEE e WHERE s.vehicle_id = v.vehicle_id AND s.servicedate = '05-OCT-13' AND s.employee_id = e.employee_id AND e.firstname = 'Joe' AND e.lastname = 'Biden' ORDER BY v.model_id;

<table>
<thead>
<tr>
<th>VEHICLE_ID</th>
<th>MODEL_ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>1309011</td>
<td>AvalonHybrid13</td>
</tr>
<tr>
<td>1309015</td>
<td>FJCruiser113</td>
</tr>
<tr>
<td>1309013</td>
<td>Highlander13</td>
</tr>
</tbody>
</table>

3 rows selected
7. List all Vehicles not Sold by Maggie Flores.

CS342 SQL> edit
Wrote file afiedt.buf

`1 * SELECT v1.vehicle_id FROM JGDM_VEHICLES v1 MINUS SELECT v.vehicle_id FROM JGDM_VEHICLES v, JGDM_EMPLOYEE e WHERE v.employee_id = e.employee_id and e.efirstname = 'Maggie' and e.elastname = 'Flores'
CS342 SQL> /

VEHICLE_ID
----------
1309001
1309002
1309004
1309005
1309006
1309007
1309008
1309009
1309010
1309011
1309012
1309013
1309015
1309016
1309017
1309018
1309020
1309021
1309022
1309023

20 rows selected.

8. Select all Vehicles sold on 10/15/13.

CS342 SQL> SELECT v.vehicle_id, v.model_id, v.customer_id FROM JGDM_VEHICLES v
Where v.sale_date = '15-OCT-2013';

VEHICLE_ID MODEL_ID CUSTOMER_ID
---------- ------------------ -----------
1309001 Camry13 10011

9. List vehicles that cost over $25,000.
CS342 SQL> SELECT v.vehicle_id, v.model_id, v.aq_price FROM JGDM_VEHICLES v WHERE v.aq_price > 25000;

2

<table>
<thead>
<tr>
<th>VEHICLE_ID</th>
<th>MODEL_ID</th>
<th>AQ_PRICE</th>
</tr>
</thead>
<tbody>
<tr>
<td>1309005</td>
<td>CamryHybrid13</td>
<td>25302</td>
</tr>
<tr>
<td>1309006</td>
<td>CamryHybrid13</td>
<td>25302</td>
</tr>
<tr>
<td>1309007</td>
<td>CamryHybrid13</td>
<td>25302</td>
</tr>
<tr>
<td>1309008</td>
<td>Avalon13</td>
<td>29214</td>
</tr>
<tr>
<td>1309009</td>
<td>Avalon13</td>
<td>29214</td>
</tr>
<tr>
<td>1309010</td>
<td>AvalonHybrid13</td>
<td>33396</td>
</tr>
<tr>
<td>1309011</td>
<td>AvalonHybrid13</td>
<td>33396</td>
</tr>
<tr>
<td>1309012</td>
<td>Highlander13</td>
<td>27568</td>
</tr>
<tr>
<td>1309013</td>
<td>Highlander13</td>
<td>27568</td>
</tr>
<tr>
<td>1309014</td>
<td>Highlander13</td>
<td>27568</td>
</tr>
<tr>
<td>1309015</td>
<td>FJCruiser113</td>
<td>26707</td>
</tr>
<tr>
<td>1309016</td>
<td>FJCruiser213</td>
<td>27577</td>
</tr>
<tr>
<td>1309017</td>
<td>Sienna13</td>
<td>25905</td>
</tr>
<tr>
<td>1309020</td>
<td>Tundra13</td>
<td>30107</td>
</tr>
</tbody>
</table>

14 rows selected

10. List Vehicles that cost under $25,000.

CS342 SQL> SELECT v.vehicle_id, v.model_id, v.aq_price FROM JGDM_VEHICLES v WHERE v.aq_price < 25000;

2

<table>
<thead>
<tr>
<th>VEHICLE_ID</th>
<th>MODEL_ID</th>
<th>AQ_PRICE</th>
</tr>
</thead>
<tbody>
<tr>
<td>1309001</td>
<td>Camry13</td>
<td>21531</td>
</tr>
<tr>
<td>1309002</td>
<td>Camry13</td>
<td>21531</td>
</tr>
<tr>
<td>1309003</td>
<td>Camry13</td>
<td>21531</td>
</tr>
<tr>
<td>1309004</td>
<td>Camry13</td>
<td>22497</td>
</tr>
<tr>
<td>1309018</td>
<td>Tacoma13</td>
<td>18726</td>
</tr>
<tr>
<td>1309019</td>
<td>Tacoma13</td>
<td>20352</td>
</tr>
<tr>
<td>1309021</td>
<td>Prius13</td>
<td>23977</td>
</tr>
<tr>
<td>1309022</td>
<td>Prius13</td>
<td>19273</td>
</tr>
<tr>
<td>1309023</td>
<td>Prius13</td>
<td>19273</td>
</tr>
</tbody>
</table>

9 rows selected.
There are many methods we can use to input data into our database. It all depends on size of the data to be inputted. For example, if the user only wishes to input a single tuple, then he/she is better off by using the INSERT INTO... VALUES (...) statement. Another way to input data is to create a .sql file in which we can write down all the tuples and then run them through sql plus. Additionally, another important aspect is the exportation of data from a database. In order, to make it simple to report data or import it to another database, one of the methods is xml. An xml file easily can transport and store data.

One of the methods to input data, was provided by our professor Dr. Wang, which is to use the java data loader tool. The tool is very useful, for it makes it very easy to insert data, unlike the single INSERT statements. What the program does is that it takes a formatted table with columns and entries, and analyzes the text and automatically creates the INSERT statement.

Ex.

```
SQL> DESC JGDM_VEHICLETYPE;

MODEL_ID          VARHCAR(30)       NOT NULL,
YEAR                    INT                          NOT NULL,
TYPEOFCAR         VARCHAR(10)        NOT NULL,
MSRP                   FLOAT                       NOT NULL

--So you would insert the data as follows:

SQL> INSERT INTO JGDM_VEHICLETYPE VALUES('CAMRY13', 2013, 4DR, 21597);
```
Phase IV

12. Common Features in Oracle PL/SQL and MS Trans-SQL

12.1.1. Components that consist of PL/SQL and Trans-SQL

Oracle PL/SQL is considered to be a structured language where functions, anonymous blocks and procedures and the basic blocks. Blocks can be defined with other blocks, which also consist of three parts, which are constraints, cursors, and user-defined variables/expressions. There are exceptions when handling a pre-defined or user-defined error that deals with the PL/SQL program.

12.1.2. Purpose of Stored Subprograms

Stored procedures in a subroutine are normally available to applications that access a relational database system. The DBMS server stores stored procedure. They are used to process data authentication or to gain entry to control mechanisms. Stored procedures are combined and unified logic; originally it was implemented in applications. With that being said, stored procedures are receive variables, they return results or modify variables and return them depending on where and how the variable is stated.

12.1.3. Benefits of calling stored Subprogram

Database programming is needed by several applications; because it is needed to be stored at the server and raised by any of the application programs. Starting a program at the server reduces the data transfer along with communication cost between the client and server when necessary. Procedures enhance the modeling power providing views by allowing more multifaceted types of derived data to be made available. Constraints that are beyond specification power of assertions and triggers, while comparing it with dynamic SQL normally store programs and remove overhead, avoid network traffic, protection against SQL injection attacks.
13. Oracle PL/SQL

13.1.1. PL/SQL Program Structure

A program structure is displayed as a block which consists of three parts, which are the
declaration that declare variables, constraints, cursors, and use-defined expressions. The
executable, which consists of SQL/SQLPLUS statements, exception handling that, has a
predefined or user-defined warning or error handled by the PL/SQL program, and the control
statement, which consists of conditional, iterative, and sequential controls.

- Conditional Controls:

  If condition THEN
  Statements;
  END IF
  IF condition THEN
  Statements;
  ELSEIF condition THEN
  Statements;
  ELSE
  Statements;
  END IF;
  EXIT-WHEN condition;

- Iterative Controls:

  LOOP
  Statements;
  END LOOP
  FOR I IN lowerbound .. upperbound LOOP
  Statements;
  END LOOP;
  For cursor_variable IN cursor_name LOOP
  Statements;
Database programmers to process individual rows returned by database system queries use cursors. Cursors enable manipulation of whole result sets at once—a capability that most procedural programming languages lack. In this scenario, a cursor enables the rows in a result-set to be processed sequentially.

Syntax:

DECLARE
CURSOR cursor_name [(parameter_name TYPE [, parameter_name TYPE])] IS select_statement;

13.1.2. Stored Procedure and Syntax of creating a stored procedure

A stored procedure is a saved section of code that handles a specific task, which must be repeated regularly.

Creating stored procedure syntax is:

CREATE [OR REPLACE] PROCEDURE procedure_name ( [parameters] ) IS [declare any variables] BEGIN
13.1.3. Stored Function and Syntax of creating Stored Function

A stored function is a stored procedure that returns a value where a procedure does not.

Creating a function syntax is:

```sql
CREATE [OR REPLACE] FUNCTION function_name ( [parameters] )
RETURN [specify datatype to return]
IS
[declare any variables]
BEGIN
[enter procedural code]
EXCEPTION
[handle any exceptions]
END;
```

13.1.4. Package and Syntax of creating a package

A package in PL/SQL is a schema object, that contains many different types, procedures, functions which can also be shared by many users who may need them for various applications in the same company.

- Syntax for a package is:

```sql
CREATE [OR REPLACE] PACKAGE package_name
[AUTHID {CURRENT_USER | DEFINER}]
{IS | AS}
```
13.1.5. Trigger and Syntax of creating a Trigger

A trigger in PL/SQL is a section of code that is executed and automatically done when a statement such as delete or update is inserted.

- Syntax for a trigger is:

```
CREATE [OR REPLACE ] TRIGGER trigger_name
{BEFORE | AFTER | INSTEAD OF }
{INSERT [OR] | UPDATE [OR] | DELETE}
[OF col_name]
```
14. Oracle PL/SQL Subprogram

14.1 Stored Procedures

A set of Subprograms are created to better and easily execute similar transactions. The subprograms included in this project are 6 stored procedures and 3 triggers. Where the stored procedures have been designed to insert and delete tuples. The triggers have been designed to record the history of deleted or updated tuples from each table into log_tables.

14.1.1 JGDM_NEWCUSTOMER

This stored procedure is designed to create a new customer. An exception is raised if the user forgot to fill in one of the entries.

CREATE OR REPLACE PROCEDURE JGDM_NEWCUSTOMER
("customer_id" IN NUMBER,
"cname" IN VARCHAR2,
"caddress" IN VARCHAR2,
"ccity" IN VARCHAR2,
"cstate" IN VARCHAR2,
"czip" IN NUMBER,
"phone" IN NUMBER
)
AS
BEGIN
    INSERT INTO JGDM_CUSTOMER VALUES ("customer_id", "cname", "caddress", "ccity", "cstate", "czip", "phone");
EXCEPTION
    WHEN OTHERS THEN
14.1.2 JGDM_NEWSERVICETYPE

This stored procedure creates a new service performed on a vehicle.

CREATE OR REPLACE PROCEDURE jgdm_newservicetype
{
    servicename  IN NUMBER,
    servicedate  IN DATE,
    result       IN VARCHAR2,
    vehicle_id   IN NUMBER,
    employee_id  IN NUMBER
}
AS service_id NUMBER;
BEGIN
    service_id := vehicle_id || 0 || 1;
    INSERT INTO jgdm_servicetype VALUES(service_id, servicename, servicedate, result, vehicle_id, employee_id);
    COMMIT;
EXCEPTION
    WHEN NO_DATA_FOUND THEN
        dbms_output.put_line('Please insert vehicle first');
    WHEN OTHERS THEN
        raise_application_error(-20999, 'An error occured in' || SQLCODE || '-ERROR-' || SQLERRM);
END jgdm_newservicetype;
/

14.1.3 JGDM_NEWVEHICLES

This stored procedure add a new vehicle into the database.

CREATE OR REPLACE PROCEDURE jgdm_newvehicles
{
    model_id      IN VARCHAR2,
    color         IN VARCHAR2,
    vin           IN VARCHAR2,
    aq_date       IN DATE,
    aq_price      IN FLOAT,
    miles         IN NUMBER,
    customer_id   IN NUMBER,
    employee_id   IN NUMBER,
    manufacturer_id IN NUMBER,

CREATE OR REPLACE PROCEDURE jgdm_newvehicles
(
    vehicle_id       NUMBER;
)
BEGIN
    INSERT INTO jgdm_vehicles
    VALUES(vehicle_id, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL);
    COMMIT;
    EXCEPTION
    WHEN NO_DATA_FOUND THEN
        dbms_output.put_line('Missing Data');
    WHEN OTHERS THEN
        raise_application_error (-20999, 'An error occurred in' || SQLCODE || '-' || SQLERRM);
END jgdm_newvehicles;
/

14.1.4 JGDM_DELETECUSTOMER

CREATE OR REPLACE PROCEDURE jgdm_deletecustomer
(
    cid       IN NUMBER
)
AS
    vid       NUMBER;
BEGIN
    SELECT vehicle_id
    INTO   vid
    FROM    jgdm_vehicles
    WHERE jgdm_vehicles.customer_id = cid;
    UPDATE jgdm_vehicles
    SET customer_id = NULL
    WHERE vehicle_id = vid;
    UPDATE jgdm_vehicles
    SET employee_id = NULL
    WHERE vehicle_id = vid;
    UPDATE jgdm_vehicles
    SET sale_date = NULL
    WHERE vehicle_id = vid;
    UPDATE jgdm_vehicles
    SET sale_price = NULL
    WHERE vehicle_id = vid;
    UPDATE jgdm_vehicles
    SET commission = NULL
    WHERE vehicle_id = vid;
DELETE FROM JGDM_CUSTOMER WHERE JGDM_CUSTOMER.customer_id = cid;
COMMIT;
EXCEPTION
  WHEN OTHERS THEN
   raise_application_error(-20999, 'An error occured in' || SQLCODE || '-' || SQLERRM);
END jgdm_deletecustomer;
/
14.1.5 JGDM_DELETESERVICETYPE

CREATE OR REPLACE PROCEDURE jgdm_deleteservicetype
(  id       IN NUMBER
)
AS
  sid      NUMBER;
BEGIN
  SELECT service_id INTO sid FROM JGDM_VEHICLES WHERE JGDM_VEHICLES.vehicle_id = sid;
  UPDATE JGDM_VEHICLES SET service_id = NULL WHERE service_id = id;
  DELETE FROM JGDM_SERVICETYPE WHERE JGDM_SERVICETYPE.service_id = id;
  COMMIT;
EXCEPTION
  WHEN OTHERS THEN
   raise_application_error(-20999, 'An error occured in' || SQLCODE || '-' || SQLERRM);
END jgdm_deleteservicetype;
/

14.1.6 JGDM_DELETEVEHICLES

CREATE OR REPLACE Procedure jgdm_deletevehicles
(  vid      IN NUMBER
)
AS
  sid      NUMBER;
BEGIN

    SELECT service_id
    INTO sid
    FROM jgdm_servicetype
    WHERE jgdm_servicetype.vehicle_id = vid;

    DELETE FROM JGDM_SERVICETYPE WHERE JGDM_SERVICETYPE.service_id = sid;
    DELETE FROM JGDM_VEHICLES WHERE JGDM_VEHICLES.vehicle_id = vid;

    COMMIT;

EXCEPTION
    WHEN OTHERS THEN
        raise_application_error(-20999, 'An error occured in' || SQLCODE || '-ERROR-' || SQLERRM);

END jgdm_deletevehicles;
/

14.1.7 JGDM_DELETEEMPLOYEE

CREATE OR REPLACE PROCEDURE jgdm_deleteemployee
(
    Eid IN NUMBER
) AS
BEGIN
    DELETE FROM JGDM_EMPLOYEE WHERE JGDM_EMPLOYEE.employee_id = eid;

    COMMIT;

EXCEPTION
    WHEN OTHERS THEN
        raise_application_error(-20999, 'An error occurred in' || SQLCODE || '-ERROR-' || SQLERRM);

END jgdm_deleteemployee
/

14.2 Triggers

These triggers are designed to record the history of the data, by moving records that have been updated or deleted into log tables.
### 14.2.1 JGDM_CUSTOMER_TRIGGER

```
CREATE OR REPLACE TRIGGER jgdm_customer_trigger
AFTER UPDATE OR DELETE ON JGDM_CUSTOMER
FOR EACH ROW
BEGIN
    INSERT INTO JGDM_CUSTOMER_LOG
    VALUES(:old.customer_id, :old.cname, :old.caddress, :old.ccity, :old.cstate,
    :old.czip, :old.phone, SYSDATE);
END jgdm_customer_trigger;
/
```

### 14.2.2 JGDM_SERVICETYPE_TRIGGER

```
CREATE OR REPLACE TRIGGER jgdm_servicetype_trigger
AFTER UPDATE OR DELETE ON JGDM_SERVICETYPE
FOR EACH ROW
BEGIN
    INSERT INTO JGDM_SERVICETYPE_LOG
    VALUES(:old.service_id, :old.servicename, :old.servicedate, :old.result,
    :old.vehicle_id, :old.employee_id, SYSDATE);
END jgdm_servicetype_trigger;
/
```

### 14.2.3 JGDM_VEHICLES_TRIGGER

```
CREATE OR REPLACE TRIGGER jgdm_vehicles_log
AFTER UPDATE OR DELETE ON JGDM_VEHICLES
FOR EACH ROW
BEGIN
    INSERT INTO JGDM_VEHICLES_LOG
    VALUES(:old.vehicle_id, :old.model_id, :old.color, :old.vin, :old.aq_date,
    :old.aq_price, :old.miles, :old.customer_id, :old.employee_id, :old.manufacturer_id,
    :old.sale_date, :old.sale_price, :old.commission, :old.service_id, SYSDATE);
END jgdm_vehicles_log;
/
```

### 14.2.4 JGDM_EMPLOYEE_TRIGGER

```
CREATE OR REPLACE TRIGGER jgdm_employee_log
AFTER UPDATE OR DELETE ON JGDM_EMPLOYEE
FOR EACH ROW
BEGIN
    INSERT INTO JGDM_EMPLOYEE_LOG
```

END jgdm_EMPLOYEE_log;
/

14.2.5 JGDM_MANUFACTURER_TRIGGER

CREATE OR REPLACE TRIGGER jgdm_manufacturer_log
    AFTER UPDATE OR DELETE ON JGDM_MANUFACTURER
    FOR EACH ROW
BEGIN
    INSERT INTO JGDM_MANUFACTURER_LOG
    VALUES(:old.manufacturer_id, :old.name, :old.address, :old.city, :old.state, :old.zip, :old.phone, SYSDATE);

END jgdm_manufacturer_log;
/
Phase V

15. DAILY ACTIVITIES OF USER GROUP

The Database is used daily by the Employees of the Sales Department in a local Toyota Dealership, in which they are able to maintain records of vehicle inventory, sales, services, customers, and employees. The Salespersons will have the ability to log into the database and search vehicle inventory that they have available for sale to show customers what are the possible purchases they can make.

The main goal of our project was to design an application that can cover a portion of the car dealerships management as well as the placement of orders made by the customers associated corresponding to what they want, and are searching for. Our graphical user interface was designed to make it easy to insert, delete, modify, and search for data. The sequential steps to use the GUI are as follows: login to GUI, view the data, insert/delete data, log data, and create the report.

16. RELATIONS, VIEWS, AND SUBPROGRAMS

There are a few tables that are a key aspect of the implementation of our program. The all connect and share 1:N relationship with an important one being jgdm_vehicle with it having many items associated with it such as jgdm_customer.

- Jgdm_vehicles: main table in the database, with foreign keys relating all over
- Jgdm_customer: main attraction to our GUI in order for the business to run properly
- Jgdm_servicetype: main relation to the vehicle_id, in order for processes to go through

A trigger for servicetype was created to keep a history of the changed data.

```
CREATE OR REPLACE TRIGGER jgdm_servicetype_trigger
AFTER UPDATE OR DELETE ON JGDM_SERVICETYPE
FOR EACH ROW
BEGIN
    INSERT INTO JGDM_SERVICETYPE_LOG
    VALUES
    (:old.service_id, :old.servicename, :old.servicedate,
    :old.result, :old.vehicle_id, :old.employee_id, SYSDATE);
```
With this example being presented, this information was provided in our previous phase.

For the GUI, all the relations previously created in the Oracle database were satisfactory for this design.

Sales View

The sales view is the first view of the program, it is used to display all sales records in the database. The Sales view includes all information needed about a sale, the vehicle_id of the sale helps us related back to the vehicle inventory. This view is important for the dealership, so they can be aware what inventory is no longer available. In the menu portion, the salesperson has the option of the Inserting or removing data into the sales view.

Vehicles View

The vehicles view is the second view button, it is used to display all vehicles in inventory. This view is also important to the dealership because it holds information on vehicles, such as the acquired data and acquired price.

Service View

The services view display information on all vehicles serviced within the department. All our cars are new models, but sometimes the manufacturer might have recalls on car parts. Such as, the brake recall that Toyota had not too long ago where the brake system wasn’t functioning correctly within the new models. So, in order to be safe and ensure the safety of our cars to the customers; all our cars are inspected before available for sale. In the service view, the user may search for services done on a specific vehicle.

Customer View

The Customer view is a bit different from all the other view because this view button leads the user to another window solely meant for customers. Where the user is able to easily search customers by id or name, and insert/delete customer data.

Employee View

This view is also set apart from all the other views, the only way to access it is through the menu item “Employee.” However, only the Sales Department Manager has access to because it is password protected. He or she is the only one allowed to search and modify employee information.
New Data

The new data stored procedures were needed in order to do this process. Unfortunately, some of the stored procedures recently created were not able to be implemented because we were not able to make an appropriate call function. However, we were able to use another method for insert, where we would declare a connection string then initiate a command to write an insert command. Amazingly the new syntax work great to insert all data. In the menu next to “File” there is a menu item for each view where the user may choose to insert data or delete data.

Delete Data

For some odd reason, the only procedures available for implementation were the delete stored procedures. In which were able to make a callback to the stored procedure in oracle to delete specified data. For example, if the user would click the menu item “Sales” and select “Delete Sale” the user would be lead to another window where they would be ask input the sale_id of the sale they wish to delete.

Report

The report view is an important component of the database because it necessary to print out sales reports to customers for their own records. In the report window the user would input the sale_id of the sale report they wish to print out, or choose to print out all sales for the year. For this, process we had to use a query to select the appropriate sales. Once the report is displayed the user has the option of printing out the report or saving it as a pdf, excel, or word document. The same process is used to display a vehicles report.

Above all the tools created PL/SQL came in great use because they saved us a lot of time. With the exception of the insert procedures of course, however a lot was learned through this experience and by far this has been the most interesting class we have taken. We are looking forward to continuing learning and working much more in databases because they have proven to be very useful and beneficial.

17. SCREENSHOTS OF CODE

Here is the start of the GUI and the login for the employee’s in the sales department, the use the login information the company provides to log in and edit things such as sales, services, and customer history.
Here is the main screen of the GUI, here you can see where the employee can access the necessary tasks needed with the individual tabs such as file, vehicle record entry/removal, sales record entry/removal, the history of logs done inside the database and an option to print the report. There is also the Employee management tab that can only be accessed by the manager with his individual log password so he can add and remove customers from the database.
The each individual tabs such as vehicle, sales, and service; displays the entry tabs and drop boxes needed to complete the necessary task such as this example with the “Add Sale” entry table, here you will enter the Vehicle_ID number of the car you worked on, the Employee_ID who made the sale, which company he/she may have worked at, as well as the price that was made by the employee as well as the sales date, commission and the Sales_ID to keep record of the sale.
Finally, we present the sales history report, which is available to select from the “Report” tab along with the “Vehicle sale report”, these are available to be saved as a pdf, word, or excel file.
Major features in our GUI include that of which there are 6 different tabs with the functionality of adding, deleting, and changing data in the program. Along side those are preview buttons below the tabs that provide more interaction with program itself, seeing that the user/customer will have a simple time figuring out the program.

Eventually with having the ability to connect to Oracle and getting MS Studios up, it was easier as time went on building the GUI as C# was really easy to catch on to and our GUI became closer to being functional. Nearing the end of our project we endured more problems with Oracle and the connection to MS Studios, but with proper knowledge and knowing our code we were able to overcome these tasks. What I’ve learned through the process of building my simple application is that connecting to a database and operating the data can be very simple once the proper knowledge of the coding is acquired. Also, with the success of an application is built is in the laying out of tasks and functions that a user might need in order to get things done quickly and efficiently.

Below you will find some code snippets to insert data into the database. The following syntax was used similarly to insert data to all the tables. The below though is only of the sales.

```csharp
OracleConnection conn = new OracleConnection("DATASOURCE=DELPHIDB;PASSWORD=c3m4p2s;PERSIST SECURITY INFO=True;USER ID=CS342");
conn.Open();
```
int o = sc.ExecuteNonQuery();
MessageBox.Show(o + " :Data has Inserted");
conn.Close();
Close();

Then following Code snippet is syntax to call the delete stored procedure. The example below is only of the vehicles entity, however the syntax was similarly used in other cases.

    // create the connection
    OracleConnection conn = new OracleConnection("DATA
    SOURCE=DELPHIDB;PASSWORD=c3m4p2s;PERSIST SECURITY INFO=True;USER ID=CS342");

    // create the command for the stored procedure
    OracleCommand cmd = new OracleCommand();
    cmd.Connection = conn;
    cmd.CommandText = "JGDM_DELETEVEHICLES";
    cmd.CommandType = CommandType.StoredProcedure;

    // add the parameter specifying the vehicle for whom to delete records

    OracleString rowId;
    // execute the stored procedure
    conn.Open();
    int rowsAffected = cmd.ExecuteNonQuery();
    MessageBox.Show(rowsAffected + ":Data Deleted");
    conn.Close();
    Close();

19. STEPS TO DESIGNING AND CREATING A DATABASE

In order to get a successful and working database, it took hard work and teamwork between partners to get things done in a timely manner, here are short descriptions of each phase:

Phase 1: Collection and Analysis:

- This phase required finding a partner and begin our analysis of the possible company we wanted to base our project on and the expectations of the users along with the intended uses of the database very detailed. As well as entity descriptions, and creating our ER Diagram.

Phase 2: Comparing ER (Conceptual) Model to Relational (Logical) Model:
- Requiring us to simply convert our ER Database into a relational database, inserting various descriptions, and inserting our queries.

Phase 3: Implementation of Relational Database

- This phase simply required us to create logical and a physical database with the Oracle DBMS.

Phase 4: Logical Database Design (Data Model Mapping):

- Explaining common features in the oracle PL/SQL and MS Trans-SQL and Oracle PL/SQL code and documentation.

Phase 5: User Interface Design and Implementation:

- This process requires us to choose a specific file storage structures and access paths for the database files in order reach a good performance for the numerous database applications to deal with.

Presentation:

- During the presentation we gave a thorough walk through of the in’s and out’s of our GUI presenting the processes of which our database was based off of.

Conclusion:

- Overall this was a challenging assignment to tackle because this was the first time I was ever introduced to a DBMS, but in the end it was fun to do and I gained a lot of experience from it.

20. ENBEDDED QUESTIONS

20. ENBEDDED QUESTIONS – OUTCOME AND MEMBERS ANSWERS

(3b) An ability to analyze a problem, and identify and define the computing requirements and specifications appropriate to its solution.

- MEMBER 1: JOHN GONZALEZ - 9
- MEMBER 2: DWAYNE MCGEE - 9

(3e) An ability to design, implement and evaluate a computer-based system, process, component, or program to meet desired needs. An ability to
understand the analysis, design, and implementation of a computerized solution to a real-life problem.

- MEMBER 1: JOHN GONZALEZ - 9
- MEMBER 2: DWAYNE MCGEE - 9

(3f) An ability to communicate effectively with a range of audiences. An ability to write a technical document such as a software specification white paper or a user manual.

- MEMBER 1: JOHN GONZALEZ - 8
- MEMBER 2: DWAYNE MCGEE - 8

(3j) An ability to apply mathematical foundations, algorithmic principles, and computer science theory in the modeling and design of computer-based systems in a way that demonstrates comprehension of the tradeoffs involved in design choices.

- MEMBER 1: JOHN GONZALEZ - 8
- MEMBER 2: DWAYNE MCGEE - 8