Phase 2 Report | CS 6400 - Fall 2021 | Team 066

Table of Contents:

Abstract Code and SQL:

Login

Abstract Code

- User type in *username* ($UserName)and *password* ($Password), click ***Log in*** button

```
SELECT Password FROM LoggedInUser WHERE UserName='$UserName';
```

- Find the user and corresponding password in the database
  - If they can match user's input:
    - Proceed to database with corresponding access
  - If they cannot match:
    - Show ***Profile not matching our record***
  - if ***Cancel*** button is clicked:
    - return to previous page

Enter Customer Information

Abstract Code

- If the salespeople or Roland selects the type of customer from a drop-down list, and click ***Add customer*** button (Section 1):
  - If salespeople or Roland selects "person customer":
    - then if he/she input a valid combination of *Address* ($Address), *City* ($City), *State* ($State), *PostalCode* ($PostalCode), *PhoneNumber* ($PhoneNumber), *EmailAddress* ($EmailAddress), *DriverLicenseNumber* ($DriverLicenseNumber), *FirstName* ($FirstName), *LastName* ($LastName):
      - When ***Enter*** button is clicked:
        - a person record will be added into Person table

```
INSERT INTO Person ('$DriverLicenseNumber', '$FirstName', '$LastName') VALUES
(DriverLicenseNumber, FirstName, LastName);
```

        - a customer record will be added into Customer table

```
INSERT INTO Customer ('$Address', '$City', '$State', '$PostalCode', '$PhoneNumber', '$EmailAddress')
VALUES (Address, City, State, PostalCode, PhoneNumber, EmailAddress);
```

      - If ***Go Back*** is clicked:
        - Go back to the Section 1
  - If salespeople or Roland selects "business customer":
    - then if he/she input a valid combination of *Address* ($Address), *City* ($City), *State* ($State), *PostalCode* ($PostalCode), *PhoneNumber* ($PhoneNumber), *EmailAddress*

($EmailAddress), *TaxIdentificationNumber* ($TaxIdentificationNumber), *BusinessName* ($BusinessName), *Name* ($Name), *Title* ($Title):

- When ***Enter*** button is clicked:
  - a business record will be added into Business table

```
INSERT INTO Business ('$TaxIdentificationNumber', '$BusinessName', '$Name', '$Title') VALUES (TaxIdentificationNumber, BusinessName, Name, Title);
```

  - a customer record will be added into Customer table

```
INSERT INTO Customer ('$Address', '$City', '$State', '$PostalCode', '$PhoneNumber', '$EmailAddress') VALUES (Address, City, State, PostalCode, PhoneNumber, EmailAddress);
```

- If ***Go Back*** is clicked:
  - Go back to the Section 1

Update Manufacturer List

Abstract Code

- If Roland clicks "Update Manufacturer List"(section 1):
  - if Roland clicks ***Delete Manufacturer***, he will see a dropdown list of current manufacturers:
    - If he selects one manufacturer ($manufacturer) from the dropdown list and click ***Delete Manufacturer***, the selected manufacture will be deleted from manufacture table

```
DELETE FROM Manufacturer WHERE Name = '$manufacturer';
```

    - If ***Go Back*** is clicked:
      - Go back to the Section 1

  - If Roland selects ***Add Manufacturer*** and inputs a new valid Manufacturer ($manufacturer):
    - When ***Add Manufacturer*** button is clicked, the new manufacturer will be added into Manufacturer table

```
INSERT INTO Manufacturer ('$manufacturer') VALUES (Name);
```

    - If ***Go Back*** is clicked:
      - Go back to the Section 1

Enter Sale Transaction

Abstract Code

- Enter *customer ID* ($CustomerID):
    - If the customer can be found in the existing database:

```
SELECT CustomerID, EmailAddress, PhoneNumber, Address, City, State, PostalCode
FROM Customer WHERE Name='$CustomerName';
```

    - else:
        - pop up enter customer information form to add a new customer

```
INSERT INTO Customer ('$Address', '$City', '$State', '$PostalCode', '$PhoneNumber', '$EmailAddress')
VALUES (Address, City, State, PostalCode, PhoneNumber, EmailAddress);
```

- Enter vehicle *VIN*($VIN):
    - If VIN can be found in database:
        - proceed to next step
    - Else:
        - return error message
- Lookup invoice price with VIN

```
SELECT InvoicePrice FROM
(SELECT VIN, InvoicePrice FROM Truck
UNION SELECT VIN, InvoicePrice FROM Convertible
UNION SELECT VIN, InvoicePrice FROM SUV
UNION SELECT VIN, InvoicePrice FROM Van
UNION SELECT VIN, InvoicePrice FROM Car) AS V WHERE V.VIN='$VIN';
```

- Enter date and sold price
    - If ***Save*** button is clicked:
        - validate all input field, if all validated:
            - save this transaction

```
INSERT INTO Purchase ('$SoldPrice', '$PurchaseDate') VALUES (SoldPrice, PurchaseDate);
```

        - else:
            - show invalid fields
    - Else:
        - Revert
- If ***Cancel*** button is clicked:
    - return to previous page
- If ***Save for Later*** button is clicked:
    - Save current process without validating

Enter Repair Details

Abstract Code

- If *VIN*($VIN) is entered and ***Search*** button is clicked:
    - Look up corresponding vehicle and customer

```
SELECT VIN, ModelName, Color, ModelYear, InvoicePrice FROM
(SELECT VIN, ModelName, Color, ModelYear, InvoicePrice FROM Truck
UNION SELECT VIN, ModelName, Color, ModelYear, InvoicePrice FROM Convertible
UNION SELECT VIN, ModelName, Color, ModelYear, InvoicePrice FROM SUV
UNION SELECT VIN, ModelName, Color, ModelYear, InvoicePrice FROM Van
UNION SELECT VIN, ModelName, Color, ModelYear, InvoicePrice FROM Car) AS V
WHERE V.VIN='$VIN';
```

```
SELECT CustomerID, PurchaseDate, SoldPrice, SalespeopleUsername
FROM Purchase WHERE VIN='$VIN';
```

    - Show profile of both vehicle and customer
- Enter *start date*($StartDate) and *end date*($EndDate), if start date larger than end date:
    - Show error message

```
INSERT INTO Repair ('$StartDate', '$EndDate') VALUES (StartDate,EndDate);
```

    - else
        - Proceed to next step
- Lookup parts details, if corresponding parts in database:

```
SELECT PartID, Quantity, VendorName, PartNumber, Price
FROM Part WHERE PartID='$PartID';
```

    - Show the details -Else
    - Proceed to enter parts details task

```
INSERT INTO Parts ('$PartID', '$Quantity', '$VendorName', '$PartNumber', '$Price') VALUES (PartID,
Quantity, VendorName, PartNumber, Price);
```

- Enter *Labor cost* ($LaborCharges)

```
INSERT INTO Repair ('$LaborCharges') VALUES (LaborCharges);
```

- Add labor cost and parts cost and save as total cost
- If ***Save*** button is clicked,
    - Validate the input fields:
        - if all validated:
            - save the form and input to database
        - else:
            - show invalid fields

- If ***Cancel*** button is clicked:
    - Go back to previous page

Search for Vehicle

Abstract Code

- Anonymous User select *Vehicle type* ($Vtype) from drop down list, enter *Manufacturer* ($Manufacturer), *Model year* ($Myear), *Color* ($Color), *List price* ($Lprice), and *Keyword* ($Keyword) input fields.
- If user is login user, then user can enter *VIN* ($VIN) input fields
    - If data validation is successful for all input fields for Anonymous User, then:
        - When ***Enter*** button is clicked:

If $Vtype = 'Truck'

```
SELECT
VIN, ModelName, ModelYear, InvoicePrice, Description, NumRearAxles, CargoCoverType,
CargoCapacity, ManufacturerName
FROM Truck;
```

If $Vtype = 'Convertible'

```
SELECT
VIN, ModelName, ModelYear, InvoicePrice, Description, RoofType, BackSearCount, ManufacturerName
FROM Convertible;
```

If $Vtype = 'SUV'

```
SELECT
VIN, ModelName, ModelYear, InvoicePrice, Description, DriveTrainType, NumCupHolders,
ManufacturerName
FROM SUV;
```

If $Vtype = 'Van'

```
SELECT
VIN, ModelName, ModelYear, InvoicePrice, Description, HasDriveSideBackDoor, ManufacturerName
FROM Van;
```

If $Vtype = 'Car'

```
SELECT
VIN, ModelName, ModelYear, InvoicePrice, Description, NumDoors, ManufacturerName
FROM Car;
```

A vehicle with attributes *VIN*($VIN), *Vehicle type*($Vtypr), *model year*($ModelYear), *manufacturer*($manufacturer), will display.

- If data validation is successful for VIN input fields for login user, then:
  - When ***Enter*** button is clicked:

If $Vtype = 'Truck'

```
SELECT
VIN, ModelName, ModelYear, InvoicePrice, Description, NumRearAxles, CargoCoverType,
CargoCapacity, ManufacturerName
FROM Truck WHERE VIN = '$VIN';
```

If $Vtype = 'Convertible'

```
SELECT
VIN, ModelName, ModelYear, InvoicePrice, Description, RoofType,BackSearCount, ManufacturerName
FROM Convertible WHERE VIN = '$VIN';
```

If $Vtype = 'SUV'

```
SELECT
VIN, ModelName, ModelYear, InvoicePrice, Description, DriveTraintype, NumCupHolders,
ManufacturerName
FROM SUV WHERE VIN = '$VIN';
```

If $Vtype = 'Van'

```
SELECT
VIN, ModelName, ModelYear, InvoicePrice, Description, HasDriveSideBackDoor, ManufacturerName
FROM Van WHERE VIN = '$VIN';
```

If $Vtype = 'Car'

```
SELECT
VIN, ModelName, ModelYear, InvoicePrice, Description, NumDoors,ManufacturerName
FROM Car WHERE VIN = 'VIN';
```

- Else if no vehicles meet the search criteria, display no vehicle found message.

Update Vehicle Inventory

Abstract Code

- User log in:
  - Show ***Add Vehicle*** button.
- Once ***Add Vehicle*** button is clicked:
  - if the user has write access to inventory form:
    - pop up inventory form
    - User select *Vehicle type* ($Vtype) from drop down list and enter input fields corresponding to each Vehicle type selected:

If $Vtype = 'Truck'

```
INSERT INTO Truck ('$VIN', '$ModelName', '$ModelYear', '$InvoicePrice', '$Description',
'$NumRearAxles', '$CargoCoverType', '$CargoCapacity', '$ManufacturerName') VALUES (VIN,
ModelName, ModelYear, InvoicePrice, Description, NumRearAxles, CargoCoverType, CargoCapacity,
ManufacturerName);
```

If $Vtype = 'Convertible'

```
INSERT INTO Convertible ('$VIN', '$ModelName', '$ModelYear', '$InvoicePrice', '$Description',
'$RoofType', '$BackSearCount', '$ManufacturerName') VALUES (VIN, ModelName, ModelYear,
InvoicePrice, Description, RoofType,BackSearCount, ManufacturerName );
```

If $Vtype = 'SUV'

```
INSERT INTO SUV ('$VIN', '$ModelName', '$ModelYear', '$InvoicePrice', '$Description',
'$DriveTrainType', '$NumCupHolders', '$ManufacturerName') VALUES (VIN, ModelName, ModelYear,
InvoicePrice, Description, DriveTrainType, NumCupHolders, ManufacturerName);
```

If $Vtype = 'Van'

```
INSERT INTO Van ('$VIN', '$ModelName', '$ModelYear', '$InvoicePrice', '$Description',
'$HasDriveSideBackDoor', '$ManufacturerName') VALUES (VIN, ModelName, ModelYear, InvoicePrice,
Description, HasDriveSideBackDoor,ManufacturerName);
```

If $Vtype = 'Car'

```
INSERT INTO Car ('$VIN', '$ModelName', '$ModelYear', '$InvoicePrice', '$Description', '$NumDoors',
'$ManufacturerName') VALUES (VIN, ModelName, ModelYear, InvoicePrice, Description,
NumDoors,ManufacturerName);
```

    - if the ***Save*** button is clicked:
      - Validate all input fields, if all valid:
        - store the input to data base, show ***Successful Added***
      - Else:
        - Show the invalid fields with hint

- Else the user does not have access:
  - Display error message, return to previous page

Get Vehicle Details

Abstract Code

Once ***Open repair form*** button is clicked:

- if the user is service writer or Roland:
  - then user can enter *VIN* ($VIN) input fields
- If data validation is successful for VIN input fields:
  - Display the selected Vehicle information

```
SELECT VIN, Color, ModelName, ModelYear, InvoicePrice, Description FROM
(SELECT VIN, ModelName, Color, ModelYear, InvoicePrice FROM Truck
UNION SELECT VIN, ModelName, Color, ModelYear, InvoicePrice FROM Convertible
UNION SELECT VIN, ModelName, Color, ModelYear, InvoicePrice FROM SUV
UNION SELECT VIN, ModelName, Color, ModelYear, InvoicePrice FROM Van
UNION SELECT VIN, ModelName, Color, ModelYear, InvoicePrice FROM Car) AS V
WHERE VIN='$VIN';
```

Enter Parts Details

Abstract Code

- Once ***Add Parts*** button is clicked:
  - if the user is service writer or Roland:
    - write in all details about the part
    - if all fields are validated:
      - If ***Save*** button is clicked:

```
INSERT INTO Part ('$PartID', '$Quantity', '$VendorName', '$PartNumber', '$Price') VALUES
( PartID ,Quantity, VendorName, PartNumber, Price);
```

        - store the details in the database
    - Else:
      - Show invalid fields with hints
  - if the user does not have access:
    - show error message
    - bring the user to previous page

Generate Reports

Abstract Code

- The Owner selects the type of report from a drop-down list, and click ***Generate Report*** button:

Sales by Color

- If the Owner selects "Sales by Color":

  And if “All time” is selected:

```
SELECT Color, SUM(CASE WHEN PurchaseDate IS NOT NULL THEN 1 ELSE 0 END) AS NumCars
FROM
   (SELECT VC.VIN, Color, PurchaseDate
    FROM
      (SELECT DISTINCT V.VIN, CASE WHEN NUM > 1 THEN "multiple" ELSE Color END AS Color
       FROM
         (SELECT VIN, COUNT(Color) AS NUM FROM VehicleColor GROUP BY VIN) AS C
       RIGHT JOIN VehicleColor V
       ON C.VIN = V.VIN) AS VC
    LEFT JOIN
      (SELECT VIN, PurchaseDate FROM Purchase) AS P
    ON VC.VIN = P.VIN) AS VCP
GROUP BY Color
ORDER BY Color ASC;
```

  If “The previous year” is selected:

```
SELECT Color, SUM(CASE WHEN PurchaseDate IS NOT NULL THEN 1 ELSE 0 END) AS NumCars
FROM
   (SELECT VC.VIN, Color, PurchaseDate
    FROM
      (SELECT DISTINCT V.VIN, CASE WHEN NUM > 1 THEN "multiple" ELSE Color END AS Color
       FROM
         (SELECT VIN, COUNT(Color) AS NUM FROM VehicleColor GROUP BY VIN) AS C
       RIGHT JOIN VehicleColor V
       ON C.VIN = V.VIN) AS VC
    LEFT JOIN
      (SELECT VIN, PurchaseDate FROM Purchase
       HAVING DATEDIFF((SELECT MAX(PurchaseDate) FROM Purchase), PurchaseDate) < 365) AS P
    ON VC.VIN = P.VIN) AS VCP
GROUP BY Color
ORDER BY Color ASC;
```

If “The previous 30 days” is selected:

```
SELECT Color, SUM(CASE WHEN PurchaseDate IS NOT NULL THEN 1 ELSE 0 END) AS NumCars
FROM
   (SELECT VC.VIN, Color, PurchaseDate
    FROM
      (SELECT DISTINCT V.VIN, CASE WHEN NUM > 1 THEN "multiple" ELSE Color END AS Color
       FROM
         (SELECT VIN, COUNT(Color) AS NUM FROM VehicleColor GROUP BY VIN) AS C
       RIGHT JOIN VehicleColor V
       ON C.VIN = V.VIN) AS VC
    LEFT JOIN
      (SELECT VIN, PurchaseDate FROM Purchase
       HAVING DATEDIFF((SELECT MAX(PurchaseDate) FROM Purchase), PurchaseDate) < 30) AS P
    ON VC.VIN = P.VIN) AS VCP
GROUP BY Color
ORDER BY Color ASC;
```

Sales by Manufacturer

- If the Owner selects "Sales by Manufacturer":

  And if “All time” is selected:

```
SELECT
   ManufacturerName, COUNT(ManufacturerName) AS NumCars
FROM
   (SELECT VM.VIN, ManufacturerName, PurchaseDate FROM
      (SELECT VIN, ManufacturerName FROM Truck
       UNION SELECT VIN, ManufacturerName FROM Convertible
       UNION SELECT VIN, ManufacturerName FROM SUV
       UNION SELECT VIN, ManufacturerName FROM Van
       UNION SELECT VIN, ManufacturerName FROM Car) AS VM
    INNER JOIN
      (SELECT VIN, PurchaseDate FROM Purchase) AS P
    ON VM.VIN = P.VIN) AS VMP
GROUP BY ManufacturerName
ORDER BY ManufacturerName ASC;
```

If “The previous year” is selected:

```
SELECT
   ManufacturerName, COUNT(ManufacturerName) AS NumCars
FROM
   (SELECT VM.VIN, ManufacturerName, PurchaseDate FROM
      (SELECT VIN, ManufacturerName FROM Truck
       UNION SELECT VIN, ManufacturerName FROM Convertible
       UNION SELECT VIN, ManufacturerName FROM SUV
       UNION SELECT VIN, ManufacturerName FROM Van
       UNION SELECT VIN, ManufacturerName FROM Car) AS VM
    INNER JOIN
      (SELECT VIN, PurchaseDate FROM Purchase
       HAVING DATEDIFF((SELECT MAX(PurchaseDate) FROM Purchase), PurchaseDate) < 365) AS P
    ON VM.VIN = P.VIN) AS VMP
GROUP BY ManufacturerName
ORDER BY ManufacturerName ASC;
```

If “The previous 30 days” is selected:

```
SELECT
   ManufacturerName, COUNT(ManufacturerName) AS NumCars
FROM
   (SELECT VM.VIN, ManufacturerName, PurchaseDate FROM
      (SELECT VIN, ManufacturerName FROM Truck
       UNION SELECT VIN, ManufacturerName FROM Convertible
       UNION SELECT VIN, ManufacturerName FROM SUV
       UNION SELECT VIN, ManufacturerName FROM Van
       UNION SELECT VIN, ManufacturerName FROM Car) AS VM
    INNER JOIN
      (SELECT VIN, PurchaseDate FROM Purchase
       HAVING DATEDIFF((SELECT MAX(PurchaseDate) FROM Purchase), PurchaseDate) < 30) AS P
    ON VM.VIN = P.VIN) AS VMP
GROUP BY ManufacturerName
ORDER BY ManufacturerName ASC;
```

Sales by Type

- If the Owner selects "Sales by Type":

  And if “All time” is selected:

```
SELECT
   VehicleType, SUM(CASE WHEN PurchaseDate IS NOT NULL THEN 1 ELSE 0 END) AS NumCars
FROM
   (SELECT VT.VIN, VehicleType, PurchaseDate FROM
      (SELECT VIN, "Truck" AS VehicleType FROM Truck
           UNION SELECT VIN, "Convertible" AS VehicleType FROM Convertible
           UNION SELECT VIN, "SUV" AS VehicleType FROM SUV
           UNION SELECT VIN, "Van" AS VehicleType FROM Van
           UNION SELECT VIN, "Car" AS VehicleType FROM Car) AS VT
    LEFT JOIN
      (SELECT VIN, PurchaseDate FROM Purchase) AS P
    ON VT.VIN = P.VIN) AS VTP
GROUP BY VehicleType
ORDER BY VehicleType ASC;
```

If “The previous year” is selected:

```
SELECT
   VehicleType, SUM(CASE WHEN PurchaseDate IS NOT NULL THEN 1 ELSE 0 END) AS NumCars
FROM
   (SELECT VT.VIN, VehicleType, PurchaseDate FROM
      (SELECT VIN, "Truck" AS VehicleType FROM Truck
           UNION SELECT VIN, "Convertible" AS VehicleType FROM Convertible
           UNION SELECT VIN, "SUV" AS VehicleType FROM SUV
           UNION SELECT VIN, "Van" AS VehicleType FROM Van
           UNION SELECT VIN, "Car" AS VehicleType FROM Car) AS VT
    LEFT JOIN
      (SELECT VIN, PurchaseDate FROM Purchase
       HAVING DATEDIFF((SELECT MAX(PurchaseDate) FROM Purchase), PurchaseDate) < 365) AS P
    ON VT.VIN = P.VIN) AS VTP
GROUP BY VehicleType
ORDER BY VehicleType ASC;
```

If “The previous 30 days” is selected:

```
SELECT
   VehicleType, SUM(CASE WHEN PurchaseDate IS NOT NULL THEN 1 ELSE 0 END) AS NumCars
FROM
   (SELECT VT.VIN, VehicleType, PurchaseDate FROM
      (SELECT VIN, "Truck" AS VehicleType FROM Truck
           UNION SELECT VIN, "Convertible" AS VehicleType FROM Convertible
           UNION SELECT VIN, "SUV" AS VehicleType FROM SUV
           UNION SELECT VIN, "Van" AS VehicleType FROM Van
           UNION SELECT VIN, "Car" AS VehicleType FROM Car) AS VT
    LEFT JOIN
      (SELECT VIN, PurchaseDate FROM Purchase
       HAVING DATEDIFF((SELECT MAX(PurchaseDate) FROM Purchase), PurchaseDate) < 30) AS P
    ON VT.VIN = P.VIN) AS VTP
GROUP BY VehicleType
ORDER BY VehicleType ASC;
```

Gross Customer Income

If the Owner selects "Gross Customer Income":

```
SELECT
   FirstName, LastName, BusinessName, FirstServiceDate, LastServiceDate,
   NumRepairs, NumSales, GrossIncome
FROM
   (SELECT
      CustomerID,
      SUM(CASE ServiceType WHEN 'Repair' THEN 1 ELSE 0 END) AS NumRepairs,
      SUM(CASE ServiceType WHEN 'Sale' THEN 1 ELSE 0 END) AS NumSales,
      MAX(ServiceDate) AS LastServiceDate,
      MIN(ServiceDate) AS FirstServiceDate,
      SUM(Income) AS GrossIncome
    FROM
      (SELECT
         CustomerID, R.StartDate AS ServiceDate,
         LaborCharges + IFNULL(PartCost,0) AS Income, 'Repair' AS ServiceType
       FROM
         (SELECT VIN, CustomerID, StartDate, LaborCharges FROM Repair) AS R
          LEFT JOIN
            (SELECT VIN, StartDate, SUM(Cost) AS PartCost
             FROM
               (SELECT PartID, Quantity * Price AS Cost FROM Part) AS P
             RIGHT JOIN Need N
             ON P.PartID = N.PartID
             GROUP BY VIN, StartDate) AS NP
          ON R.VIN = NP.VIN AND R.StartDate = NP.StartDate
       UNION
       (SELECT
         CustomerID, PurchaseDate AS ServiceDate, SoldPrice AS Income, 'Sale' AS ServiceType
        FROM Purchase)) AS PR
    GROUP BY CustomerID) AS GI
    LEFT JOIN
      (SELECT
         CP.CustomerID, FirstName, LastName, BusinessName
       FROM
         (SELECT C.CustomerID, FirstName, LastName
          FROM Customer C LEFT JOIN Person P
          ON P.CustomerID = C.CustomerID) AS CP
          LEFT JOIN Business B
          ON CP.CustomerID = B.CustomerID) AS CPB
    ON GI.CustomerID = CPB.CustomerID
 ORDER BY GrossIncome DESC, LastServiceDate DESC LIMIT 15;
```

If the customer's name is clicked, two drilldown reports will be generated and displayed:

Vehicle sales for a specific customer:

```
SELECT
   VIN, PurchaseDate, SoldPrice, ManufacturerName,
   ModelYear, ModelName, FirstName AS SalesFirstName, LastName AS SalesLastName
FROM
   (SELECT
      VIN, PurchaseDate, SoldPrice, ManufacturerName,
      ModelYear, ModelName,
      FirstName AS CustomerFirstName,
      LastName AS CustomerLastName,
      BusinessName, SalespeopleUsername
   FROM
      (SELECT
         P.VIN, CustomerID, PurchaseDate, SoldPrice,
         ModelYear, ModelName, SalespeopleUsername, ManufacturerName
      FROM
         (SELECT
            VIN, CustomerID, PurchaseDate, SoldPrice,
            SalespeopleUsername
         FROM Purchase) AS P
         LEFT JOIN
            (SELECT VIN, ModelYear, ModelName, ManufacturerName FROM Truck
            UNION SELECT VIN, ModelYear, ModelName, ManufacturerName FROM Convertible
            UNION SELECT VIN, ModelYear, ModelName, ManufacturerName FROM SUV
            UNION SELECT VIN, ModelYear, ModelName, ManufacturerName FROM Van
            UNION SELECT VIN, ModelYear, ModelName, ManufacturerName FROM Car) AS V
         ON P.VIN = V.VIN) AS VP
      LEFT JOIN
         (SELECT
            CP.CustomerID, FirstName, LastName, BusinessName
         FROM
            (SELECT C.CustomerID, FirstName, LastName
            FROM Customer C LEFT JOIN Person P
            ON P.CustomerID = C.CustomerID) AS CP
            LEFT JOIN Business B
            ON CP.CustomerID = B.CustomerID) AS CPB
      ON VP.CustomerID = CPB.CustomerID) AS VPC
   LEFT JOIN
      (SELECT S.UserName, FirstName, LastName
      FROM SalesPeople S
      INNER JOIN LoggedInUser L
      ON S.UserName = L.UserName) AS S
   ON VPC.SalespeopleUsername = S.UserName
WHERE
   (CustomerFirstName = '$CustomerFirstName' AND CustomerLastName = '$CustomerLastName') OR
   BusinessName = '$BusinessName'
ORDER BY PurchaseDate DESC, VIN ASC;
```

Repairs for a specific customer:

```
SELECT
   VIN, StartDate, DateCompleted, Odometer, LaborCharges, PartCost, TotalCost,
   FirstName AS ServiceWriterFirstName, LastName AS ServiceWriterLastName
FROM
   (SELECT
      VIN, StartDate, DateCompleted, Odometer, LaborCharges, PartCost, TotalCost, UserName,
      FirstName AS CustomerFirstName, LastName AS CustomerLastName, BusinessName
   FROM
      (SELECT
         R.VIN, R.StartDate, DateCompleted, Odometer, CustomerID, LaborCharges, PartCost,
         UserName, LaborCharges + IFNULL(PartCost,0) AS TotalCost
      FROM
         (SELECT
          VIN, StartDate, DateCompleted, Odometer, CustomerID, LaborCharges, UserName
          FROM Repair) AS R
          LEFT JOIN
            (SELECT VIN, StartDate, SUM(Cost) AS PartCost
             FROM
               (SELECT PartID, Quantity * Price AS Cost FROM Part) AS P
             RIGHT JOIN Need N
             ON P.PartID = N.PartID
             GROUP BY VIN, StartDate) AS NP
          ON R.VIN = NP.VIN AND R.StartDate = NP.StartDate) AS R
      LEFT JOIN
         (SELECT
            CP.CustomerID, FirstName, LastName, BusinessName
         FROM
            (SELECT C.CustomerID, FirstName, LastName
            FROM Customer C LEFT JOIN Person P
            ON P.CustomerID = C.CustomerID) AS CP
            LEFT JOIN Business B
            ON CP.CustomerID = B.CustomerID) AS CPB
      ON R.CustomerID = CPB.CustomerID) AS CR
   LEFT JOIN
      (SELECT S.UserName, FirstName, LastName
      FROM ServiceWriter S
      INNER JOIN LoggedInUser L
      ON S.UserName = L.UserName) AS S
   ON CR.UserName = S.UserName
WHERE
   (CustomerFirstName = '$CustomerFirstName' AND CustomerLastName = '$CustomerLastName') OR
   BusinessName = '$BusinessName'
ORDER BY DateCompleted IS NULL DESC, StartDate DESC, DateCompleted DESC, VIN ASC;
```

Repairs by Manufacturer/Type/Model

If the Owner selects "Repairs by Manufacturer":

```
SELECT
   Manufacturer,
   SUM(CASE WHEN StartDate IS NOT NULL THEN 1 ELSE 0 END) AS NumRepairs,
   SUM(LaborCharges) AS AllLaborCharges,
   SUM(PartCost) AS AllPartCost,
   SUM(TotalCost) AS AllTotalCost
FROM
   (SELECT M.Name AS Manufacturer, VIN
    FROM
      (SELECT Name FROM Manufacturer) AS M
       LEFT JOIN
         (SELECT VIN, ManufacturerName FROM Truck
         UNION SELECT VIN, ManufacturerName FROM Convertible
         UNION SELECT VIN, ManufacturerName FROM SUV
         UNION SELECT VIN, ManufacturerName FROM Van
         UNION SELECT VIN, ManufacturerName FROM Car) AS VM
       ON M.Name = VM.ManufacturerName) AS VM
   LEFT JOIN
      (SELECT
         R.VIN, R.StartDate, LaborCharges, PartCost,
         LaborCharges + IFNULL(PartCost,0) AS TotalCost
       FROM
         (SELECT VIN, StartDate, LaborCharges FROM Repair) AS R
          LEFT JOIN
            (SELECT VIN, StartDate, SUM(Cost) AS PartCost
             FROM
               (SELECT PartID, Quantity * Price AS Cost FROM Part) AS P
             RIGHT JOIN Need N
             ON P.PartID = N.PartID
             GROUP BY VIN, StartDate) AS NP
          ON R.VIN = NP.VIN AND R.StartDate = NP.StartDate) AS R
   ON VM.VIN = R.VIN
GROUP BY Manufacturer
ORDER BY Manufacturer ASC;
```

If the Owner clicks a manufacturer for drilldown, repairs by type for a specific manufacturer will display:

```
SELECT
   VehicleType,
   COUNT(*) AS NumRepairs,
   SUM(LaborCharges) AS AllLaborCharges,
   SUM(PartCost) AS AllPartCost,
   SUM(TotalCost) AS AllTotalCost
FROM
   (SELECT VIN, ManufacturerName, "Truck" AS VehicleType FROM Truck
   UNION SELECT VIN, ManufacturerName, "Convertible" AS VehicleType FROM Convertible
   UNION SELECT VIN, ManufacturerName, "SUV" AS VehicleType FROM SUV
   UNION SELECT VIN, ManufacturerName, "Van" AS VehicleType FROM Van
   UNION SELECT VIN, ManufacturerName, "Car" AS VehicleType FROM Car) AS VT
   RIGHT JOIN
      (SELECT
         R.VIN, R.StartDate, LaborCharges, PartCost,
         LaborCharges + IFNULL(PartCost,0) AS TotalCost
       FROM
         (SELECT VIN, StartDate, LaborCharges FROM Repair) AS R
          LEFT JOIN
            (SELECT VIN, StartDate, SUM(Cost) AS PartCost
             FROM
               (SELECT PartID, Quantity * Price AS Cost FROM Part) AS P
             RIGHT JOIN Need N
             ON P.PartID = N.PartID
             GROUP BY VIN, StartDate) AS NP
          ON R.VIN = NP.VIN AND R.StartDate = NP.StartDate) AS R
   ON VT.VIN = R.VIN
WHERE ManufacturerName = '$ManufacturerName'
GROUP BY VehicleType
ORDER BY VehicleType ASC;
```

If the Owner clicks a model for a further drilldown, repairs by model for a specific type and a specific manufacturer will display:

```
SELECT
   ModelName,
   COUNT(*) AS NumRepairs,
   SUM(LaborCharges) AS AllLaborCharges,
   SUM(PartCost) AS AllPartCost,
   SUM(TotalCost) AS AllTotalCost
FROM
   (SELECT VIN, ModelName, ManufacturerName, "Truck" AS VehicleType FROM Truck
   UNION SELECT VIN, ModelName, ManufacturerName, "Convertible" AS VehicleType FROM
Convertible
   UNION SELECT VIN, ModelName, ManufacturerName, "SUV" AS VehicleType FROM SUV
   UNION SELECT VIN, ModelName, ManufacturerName, "Van" AS VehicleType FROM Van
   UNION SELECT VIN, ModelName, ManufacturerName, "Car" AS VehicleType FROM Car) AS VT
   RIGHT JOIN
      (SELECT
         R.VIN, R.StartDate, LaborCharges, PartCost,
         LaborCharges + IFNULL(PartCost,0) AS TotalCost
       FROM
         (SELECT VIN, StartDate, LaborCharges FROM Repair) AS R
          LEFT JOIN
            (SELECT VIN, StartDate, SUM(Cost) AS PartCost
             FROM
               (SELECT PartID, Quantity * Price AS Cost FROM Part) AS P
             RIGHT JOIN Need N
             ON P.PartID = N.PartID
             GROUP BY VIN, StartDate) AS NP
          ON R.VIN = NP.VIN AND R.StartDate = NP.StartDate) AS R
   ON VT.VIN = R.VIN
WHERE ManufacturerName = '$ManufacturerName' AND VehicleType = '$VehicleType'
GROUP BY ModelName
ORDER BY ModelName ASC;
```

Below Cost Sales

If the Owner selects "Below Cost Sales", the "Below Cost Sales" report will display:

```
SELECT
   PurchaseDate, InvoicePrice, SoldPrice, SalespeopleUsername, Ratio,
   CustomerFirstName, CustomerLastName, BusinessName, SalesFirstName, SalesLastName
FROM
   (SELECT
      PurchaseDate, InvoicePrice, SoldPrice, SalespeopleUsername, Ratio,
      CustomerFirstName, CustomerLastName, BusinessName
    FROM
      (SELECT
         PurchaseDate, InvoicePrice, SoldPrice, CustomerID, SalespeopleUsername,
         SoldPrice/InvoicePrice AS Ratio
       FROM
         (SELECT VIN, InvoicePrice FROM Truck
          UNION SELECT VIN, InvoicePrice FROM Convertible
          UNION SELECT VIN, InvoicePrice FROM SUV
          UNION SELECT VIN, InvoicePrice FROM Van
          UNION SELECT VIN, InvoicePrice FROM Car) AS V
       INNER JOIN Purchase P ON V.VIN = P.VIN HAVING Ratio < 1) AS VP
    LEFT JOIN
      (SELECT
         CP.CustomerID, FirstName AS CustomerFirstName,
         LastName AS CustomerLastName, BusinessName
       FROM
         (SELECT C.CustomerID, FirstName, LastName
         FROM Customer C LEFT JOIN Person P
         ON P.CustomerID = C.CustomerID) CP
       LEFT JOIN Business B
       ON CP.CustomerID = B.CustomerID) AS CPB
   ON VP.CustomerID = CPB.CustomerID) AS VPC
LEFT JOIN
   (SELECT
    S.UserName, FirstName AS SalesFirstName, LastName AS SalesLastName FROM SalesPeople S
    INNER JOIN LoggedInUser L
    ON S.UserName = L.UserName) AS S
ON VPC.SalespeopleUsername = S.UserName
ORDER BY PurchaseDate DESC, Ratio DESC;
```

Average Time in Inventory

If the Owner selects "Average Time in Inventory", the "Average Time in Inventory" report will display:

```
SELECT
   VehicleType, AVG(DATEDIFF(PurchaseDate, AddDate)) + 1 AS AvgDaysInventory
FROM
   (SELECT VT.VIN, VehicleType, AddDate FROM
      (SELECT VIN, "Truck" AS VehicleType FROM Truck
           UNION SELECT VIN, "Convertible" AS VehicleType FROM Convertible
           UNION SELECT VIN, "SUV" AS VehicleType FROM SUV
           UNION SELECT VIN, "Van" AS VehicleType FROM Van
           UNION SELECT VIN, "Car" AS VehicleType FROM Car) AS VT
    LEFT JOIN
      (SELECT VIN, AddDate FROM AddVehicle) AS A
    ON VT.VIN = A.VIN) AS VTA
    LEFT JOIN
         (SELECT VIN, PurchaseDate FROM Purchase) AS P
          ON VTA.VIN = P.VIN
GROUP BY VehicleType
ORDER BY VehicleType ASC;
```

Parts Statistics

If the Owner selects "Parts Statistics", the "Parts Statistics" report will display:

```
SELECT VendorName, SUM(Quantity), SUM(Quantity * Price) AS TotalCost
FROM Part
GROUP BY VendorName
ORDER BY TotalCost DESC;
```

Monthly Sales

If the Owner selects "Monthly Sales", the " Monthly Sales" report will display:

```
SELECT
   YEAR(PurchaseDate) AS Year, MONTH(PurchaseDate) AS Month,
   COUNT(*) AS NumSold,
   SUM(SoldPrice) AS SalesIncome, (SUM(SoldPrice) - SUM(InvoicePrice)) AS NetIncome,
   (SUM(SoldPrice) / SUM(InvoicePrice)) AS Ratio
FROM
   (SELECT VIN, PurchaseDate, SoldPrice FROM Purchase) AS P
   INNER JOIN
      (SELECT VIN, InvoicePrice FROM Truck
       UNION SELECT VIN, InvoicePrice FROM Convertible
       UNION SELECT VIN, InvoicePrice FROM SUV
       UNION SELECT VIN, InvoicePrice FROM Van
       UNION SELECT VIN, InvoicePrice FROM Car) AS V
   ON P.VIN = V.VIN
GROUP BY Year, Month
ORDER BY Year DESC, Month DESC;
```

If a year and a month is selected, a drilldown report will display for top performing salespeople:

```
SELECT
   FirstName, LastName, COUNT(*) AS NumSold, SUM(SoldPrice) AS TotalSales
FROM
   (SELECT VIN, PurchaseDate, SoldPrice, SalespeopleUsername
    FROM Purchase
    WHERE YEAR(PurchaseDate) = '$Year' AND MONTH(PurchaseDate) = '$Month') AS P
    INNER JOIN
      (SELECT S.UserName, FirstName, LastName
       FROM SalesPeople S
       INNER JOIN LoggedInUser L
       ON S.UserName = L.UserName) AS S
    ON P.SalespeopleUsername = S.UserName
GROUP BY FirstName, LastName
ORDER BY NumSold DESC, TotalSales DESC;
```

- Upon:
    - Click ***Search for Vehicles*** button - Jump to the **Search for Vehicle** task.
    - Click ***Add a Vehicle*** button - Jump to the **Add Vehicle** task.
    - Click ***Sell a Vehicle*** button - Jump to the **Sell Vehicle** task.
    - Click ***Add a Repair*** button - Jump to the **Add Repair** task.
    - Click ***Log out*** button - Invalidate the login session and go to the **Search for Vehicle** form.