

# Phase 1 Report | CS 6400 - Fall 2021 | Team 022

## Contents

- Data Type ..... 2
- Business Logic Constraints ..... 4
- Task Decomposition and Abstract Code ..... 6
  - Main Menu ..... 6
  - Login..... 7
  - Search Vehicle..... 7
  - Add Vehicle..... 8
  - Search/Add Customer..... 9
  - Add Sales..... 10
  - View/Add/Update Repair ..... 11
  - View Sales by Color Report..... 12
  - View Sales by Type Report..... 13
  - View Sales by Manufacturer Report ..... 14
  - View Gross Customer Income Report..... 14
  - View Repairs by Manufacturer/Type/Model Report..... 16
  - View Below Cost Sales Report ..... 17
  - View Average Time in Inventory Report..... 17

## Data Type

### Privileged Users

Attribute	Data type	Nullable
Username	String	Not Null
Password	String	Not Null
First Name	String	Not Null
Last Name	String	Not Null
Roles	list<String>	Not Null

Format

### Vehicles

Attribute	Data type	Nullable
VIN	String	Not Null
Vehicle type	list<String>	Not Null
Manufacturer name	list<String>	Not Null
Model name	String	Not Null
Model Year	Date	Not Null
Date added	Date	Not Null
Invoice Price	Float	Not Null
Sold	Boolean	Not Null
Description	String	NULL
List price	Float	Not Null
Color	list<String>	Not Null

Format
Alphanumeric, 17 digits without I, O, Q
[YYYY]
[MM-DD-YYYY]
two digits after decimal point, positive
[Sold, Unsold]
two digits after decimal point, positive

### Customer- Persons

Attribute	Data type	Nullable
Address	String	Not Null
Phone number	String	Not Null
Email	String	NULL
Fist Name	String	Not Null
Last Name	String	Not Null
Driver's license number	String	Not Null

Format
Valid address format
Numeric, 10 or 11 digits
Valid email format: such as username@aa.bb

### Customer- Business

Attribute	Data type	Nullable
Address	String	Not Null
Phone number	String	Not Null
Email	String	NULL

Format
Valid address format
Numeric, 10 or 11 digits
Valid email format: such as username@aa.bb

Tax Identification ID	String	Not Null
Business name	String	Not Null
Primary Contact's Name	String	Not Null
Title of primary contact	String	Not Null

Numeric, 9 digits

### Sales Events

Attribute	Data type	Nullable
Sold price	Float	Not Null
Sale Date	Date	Not Null

Format
two digits after decimal point, positive
[MM-DD-YYYY]

### Repair Events

Attribute	Data type	Nullable
Labor Charge	Float	Not Null
Start Date	Date	Not Null
End Day	Date	Null
Odometer	Integer	Not Null
Description	String	Not Null
Total Cost	Float	Not Null
Status	Boolean	Not Null
Part Cost	Float	
Description	String	

Format
two digits after decimal point, non-negative
[MM-DD-YYYY]
[MM-DD-YYYY]
Positive
two digits after decimal point, non-negative
[Open, Close]
two digits after decimal point, positive

### Needs Relationship

Attribute	Data type	Nullable
Quantity used	Integer	Not Null

Format
Positive

### Parts

Attribute	Data type	Nullable
Price	Float	Not Null
Part Number	String	Not Null

Format
two digits after decimal point, non-negative

### Cars

Attribute	Data type	Nullable
Number of Doors	Integer	Not Null

Nullable
Positive

### Van/Minivans

Attribute	Data type	Nullable
Has Driver's side back door	Boolean	Not Null

Format
[Yes, No]

### SUVs

Attribute	Data type	Nullable
Drivetrain type	String	Not Null
Number of cupholders	Integer	Not Null

Format
Non-negative

### Convertibles

Attribute	Data type	Nullable
Roof type	String	Not Null
Back seat count	Integer	Not Null

Format
Non-negative

### Trucks

Attribute	Data type	Nullable
Number of rear axles	Integer	Not Null
Cargo capacity	float	Not Null
Cargo cover type	String	Not Null

Format
Positive
Positive

## Business Logic Constraints

### Operational Constraints

- Each vehicle type must have specific attributes with valid manufacturers listed in the table in the appendix. The vehicles' color must be chosen from the table in the appendix.
- Model year of a vehicle should not be greater than current year+1
- The sale price must be greater than 95% of invoice price, except for Roland Around (owner).
- The listed price of the vehicle must be calculated as 125% of invoice price.
- Vehicles bought by customers must go through a salesperson.
- A vehicle's purchase date cannot be earlier than the date added to the inventory
- Although Roles is a multi-value attribute. The employees will have only one role. The owner, Roland Around, will have all the other roles: salesperson, inventory clerk, service writer, and manager.
- Repair services must go through a service writer
- Only sold vehicles can receive services
- A vehicle must not have more than one repair start on the same date, and a repair must be closed before a new one can be added
- When a repair is not completed, the Status of the repair is Open and the End Day is Null. When the repair is completed, the status is Close and the End Day is not Null.
- The parts cost of repair events will be determined by combining all the part cost \* quantity used
- The total cost of repair events must be determined by combining the labor costs with the cost of all parts used for the repair
- Except for the owner Roland Around, labor charges can only be changed to a higher level than the original

- There is only one owner, Roland Around.

#### Application Functionality Constraints

- All logged-in users must be identified using a unique username and a password.
- Anonymous can only access the public-facing search page for vehicles.
- Privileged users have the ability to look up and add customers to the system. It is only available when performing various transactions and not to be independently accessible.
- Besides lookup and adding customers and searching for vehicles, salespeople only have access to searching available inventory, and entering sales transactions using the sales order form.
- Besides looking up vehicles, Inventory clerks only have access to add new vehicles.
- Besides looking up vehicles, adding customers and searching for vehicles, service writers only have access to enter the details of repairs in the service writer form
- Managers have view-only access to all information along with reports.
- Reports for Roland and his managers must be via a link, button, or dropdown menu that can be displayed on the initial search page.
- Roland Around must be able to view all information and reports as manager and must be able to do any activities of salespeople, inventory clerk, and service writer.

## Task Decomposition and Abstract Code

### Markups:

**Bold uppercase initial:** task

**bold lowercase:** subtask

*Italic Bold:* button

**Underline Bold:** Form

*Italic:* Input Fields

**Blue:** Entity or Relationship

'\$Green' : variable assigned by input

"Double quoted": messages shown or assigned values

Underline: variable developed from calculation

### Main Menu

**Lock Types:** Read lock for Privileged **Users** entity

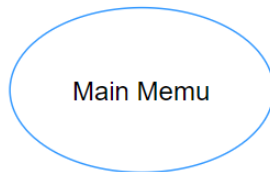
**Number of Locks:** 1

**Enabling Conditions:** Privileged User login

**Frequency:** Very high, every time when privileged users use the system

**Consistency (ACID):** Consistency not critical, order is not critical.

**Subtasks:** Mother Task is not needed. Order is not necessary.



- Display the first name, last name, and role of the Privileged **User**
- Show the *Search Vehicle* button, push to go to **Search Vehicle** form
- If the user is a **Service Writer**:
  - Show the *View/Add/Update Repair* button, push to go to **View/Add/Update Repair** task
- If the user is an **Inventory Clerk**:
  - Show the *Add Vehicle* button, push to go to **Add Vehicle** task
- If the user is a **Manager**:
  - Show the *Sales by Color Report* button to the **View Sales by Color Report** task
  - Show the *Sales by Type Report* button to the **View Sales by Type Report** task
  - Show the *Sales by Manufacturer Report* button to the **View Sales by Manufacturer Report** task
  - Show the *Gross Customer Income Report* button to the **View Gross Customer Income Report** task
  - Show the *Repairs Report* button to the **View Repairs by Manufacturer/Type/Model Report** task
  - Show the *Below Cost Sales Report* button to the **View Below Cost Sales Report** task
  - Show the *Average Time in Inventory Report* button to the **View Average Time in Inventory Report** task
- If the user is the **Owner**:
  - All buttons above will be shown
- Push *Cancel* to return to close the page

## Login

**Lock Types:** Read-only on [Privileged Users](#)

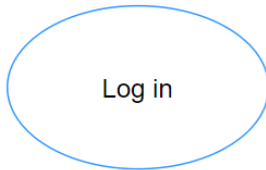
**Number of Locks:** Single

**Enabling Conditions:** None

**Frequency:** High

**Consistency (ACID):** Consistency not critical, order is not critical.

**Subtasks:** Mother Task is not needed. No decomposition needed.



- User clicks ***Sign in*** button on **Vehicle Search** form.
  - User enters *username* ('\$UserID'), *password* ('\$Password') in the input fields and clicks ***Enter*** button.
- If *email* and *password* inputs are invalid:
  - Raise message “Invalid input”
- Else, the *username* and *password* inputs are valid:
  - If User record is found but the password != '\$Password':
    - Go back to **Login** form, with an error message “Wrong password”.
  - Else:
    - Store login information as session variable '\$UserID'.
    - **Vehicle Search** Form page shows the login information
    - Go to the **Main Menu**

## Search Vehicle

**Lock Types:** Read lock for [Vehicles](#) for all users, [Repair Events](#), [Sales Events](#), [Customers](#), [Privileged Users](#) ([Salespeople](#), [Service Writers](#), [Inventory Clerks](#)) when managers and the owner using

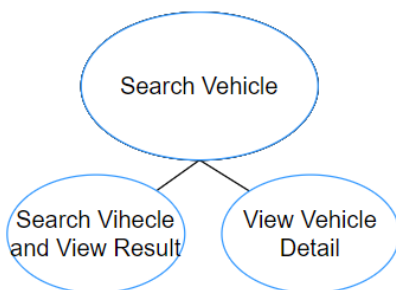
**Number of Locks:** 5

**Enabling Conditions:** None

**Frequency:** Very high, more than 500 every day.

**Consistency(ACID):** Consistency is not critical.

**Subtasks:** Decomposition is needed. Mother Task is needed to coordinate subtasks. Order is critical: Search Vehicles And View Result should be implemented first and must be done.



(Subtask: **search vehicle and view result**)

- Display the total number of unsold [Vehicles](#) available for purchase
- User optionally enters *keyword* ('\$Keyword') input field;

- User optionally enters *price* ('\$Price') input field;
- User optionally selects an option from dropdown menus for "Vehicle type", "Manufacturer", "Model year", "Color"
- Show a **Sign in** button to the **Login** task
  - If the user is a Privileged user
  - User enters VIN ('\$VIN') input field.
  - If the **Privileged User** is a **Manager** or **Owner**:
    - The user has the option to filter by sold vehicles, unsold vehicles, or all vehicles.
- Clicked the **Search** button;
- If find **Vehicles** meet the search criteria;
  - Go to the **Search Result Page** form.
    - Display VIN, Vehicle type, Model Year, Manufacturer, Model, color(s) and List price of all unsold **Vehicles** match the criteria.
    - Sort the result by VIN ascending
    - Indicate if the keyword matches the description on a check box X mark.
- If no **Vehicles** meet the search criteria;
  - An error message "Sorry, it looks like we don't have that in stock!" is displayed.
  - Go back to the **Search Vehicle** form.

(Subtask: **view vehicle details**)

- User Clicks on an individual search result and display the following information in the **Search Vehicle Detail Page** form:
  - Display VIN, vehicle type, attributes for that vehicle type, Model Year, Model Name, Manufacturer, color(s), list price, and description.
  - If user is a Salesperson or the Owner:
    - Display a **Sale Vehicle** button behind each vehicle to go to the **Sales Order** form
  - If user is a Manager or Owner:
    - Find the **Inventory Clerk** associated with the Vehicle
    - Display the Inventory Clerk's name, the invoice price, and the date added
    - If the **Vehicle** is sold:
      - Find the **Sales Event**; Get the sale date, sale price
      - Find the **Customer** associated with the Sales Event; Get the information
      - Find the **Salesperson**; Get the Salesperson's name
      - Display all the Customer information (except the Key), list price, sale price, sales date, and the Salesperson's 's name.
    - If the **Vehicle** has been repaired:
      - Find the **Repair Events**; Get the start date, end date, labor cost, part cost, and total cost
      - Find the **Customers** associated with the Repair Events; Get the Customer's name
      - Find the **Service Writer** associated with the Repair Events; Get the name
      - Show a "Repair" section, Display the Customers' name, the Service Writer's name, start date, end date, labor cost, part cost, and total cost.

## Add Vehicle

**Lock Types:** Write lock for **Vehicles**

**Number of Locks:** 1

**Enabling Conditions:** Inventory clerk or Owner login, push button on **Main Menu** form

**Frequency:** every time when a vehicle needs to be added, about 0-30 per day, medium

**Consistency (ACID):** Critical, a vehicle cannot be added by more than one user at the same time



**Subtask:** A mother task is not needed, and order is not necessary. No decomposition needed.



- Inventory clerk or Owner clicks on the **Add Vehicle** button from the **Main Menu** form
- Enter *VIN, manufacturer, model name, model year, invoice price, colors, description*
- Select *type* in the drop-down menu
  - Enter the *specific attribute* of the type
- If information is incomplete or invalid:
  - Raise message “Incomplete or invalid information, please enter again”
- Else:
  - Add a new instance in **Vehicles**
  - The sold attribute of the **Vehicle** is “unsold”
  - The list price will be 125% \* invoice price
  - Save the current date to date added
- Return to the **Add Vehicle** form

## Search/Add Customer

**Lock Types:** Read lock for the **Customers**; write lock for **Customers**

**Number of Locks:** 2

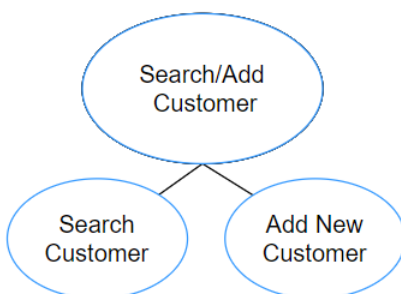
**Enabling Conditions:** Enabled by a Service Writer, Salesperson, or Owner login, pushing a button in the transaction form (**Service Writer Form** or **Sales Order Form**)

**Frequency:** Read and write will be in different frequencies. Read: high, every time in sales or repair transactions.

Write: medium, when customer not found

**Consistency (ACID):** Consistency is critical for adding new customers.

**Subtasks:** Mother Task is needed. The task decomposition is needed. Tasks do not need to be done together.



• User can only access this task by pushing button in **Service Writer Form** or **Sales Order Form**  
(Subtask: **search customer**)

- User selects **Customer** type: *person* or *business*, push **Confirm** to continue
- If person, enter *license* and push **Search**:
  - Find the **person** using the license
  - Display First Name, Last Name, License, Address, Phone Number, Email
- If business, user enter *Tax Identification Number* and push **Search**:
  - Find the **business** using the Tax Identification Number

- Display Name, Tax Identification Number, Contact Name, Contact Title, Address, Phone Number, Email
- If not found:
  - raise message “Not Found”
- Push ***Search Again*** to return to select customer type

(Subtask: **add new customer**)

- Push the ***Add New Customer*** button
- User selects **Customer** type: *person* or *business*
- If **Person**, enter:
  - *First Name, Last Name, License, Address, Phone Number, Email*
- If **Business**, enter:
  - *Name, Tax Identification Number, Contact Name, Contact Title, Address, Phone Number, Email*
- Push ***Add*** button:
  - If information is completed and valid:
    - Add a new instance of **Customer** with the information
  - Else:
    - raise message “Invalid or incomplete information”
- Push ***Confirm Customer***:
  - Return to the **Service Writer Form** or **Sales Order Form**; And display “Customer Confirmed” on the form
- Push ***Cancel*** :
  - close the form; return to the **Service Writer Form** or **Sales Order Form** without selecting a customer

## Add Sales

**Lock Types:** Write locks for **Vehicles** and **Sales Events**


**Number of Locks:** 2

**Enabling Conditions:** Enabled by a Salesperson or Owner login, Pushing a button on the **Search Vehicle Detail Page** Form

**Frequency:** Only one Frequency, frequency medium, every time sales a vehicle

**Consistency (ACID):** Consistency is critical. A sale event cannot be added by more than one user at the same time. Order is no necessary

**Subtasks:** A mother task is not needed.



Add Sales

- Salespeople push a button on **Search Vehicle Detail Page**
- Find the **Vehicle** selected in **Search Vehicle** task
- Push a **Customer** button to prompt the **Search/Add Customer** form to find **Customer**
- Enter: the *sale price* (***\$salePrice***)
- Push ***Confirm Button***
  - If the ***\$salePrice*** is not valid:
    - raise message “Invalid Entry”
  - If the ***\$salePrice*** is not higher than 95% of the invoice price, and the user is not Roland Around:

- raise message “Price too low”
- Else:
  - Update the Sold attribute of **Vehicle** from Unsold to Sold
  - Add a new **Sales Event** instance with ‘\$salePrice’
  - Save the current date as Sale date of the **Sales Event**
  - Display a message “Sale has been successfully added”
  - Close this **form**; go to the **Search Vehicle Detail Page** form
- Push **Cancel** :
  - go to the **Search Vehicle Detail Page** Form without filing the sales

## View/Add/Update Repair

**Lock Types:** Read locks for **Vehicles**, **Repair Events**, **Parts**, **Needs**; Write locks for **Repair Events** and **Needs**.

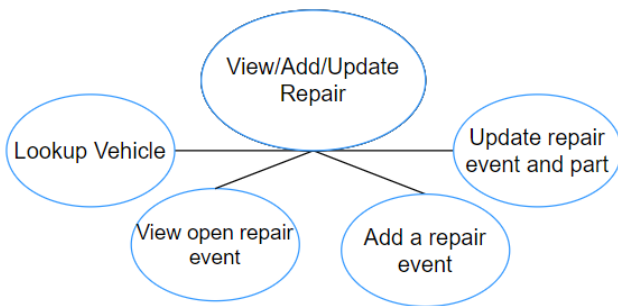
**Number of Locks:** 6

**Enabling Conditions:** Enabled by a Service Writer or owner login and push the button to open the task on the **Main Menu**

**Frequency:** Medium. Reading the **Parts** and **Needs** are in the same frequency, but all others are different.

**Consistency (ACID):** Consistency is critical. Need to ensure a Repair Event instance is not added by more than one user at the same time.

**Subtasks:** Mother Task is needed to coordinate subtasks. Order is critical, Lookup Vehicle always goes first. Subtasks do not need to be done together



- Service Writer or Owner push button **View/Add/Update Repair** on the **Main Menu**

(Subtask: **lookup vehicle**)

- User enter the **VIN** (‘\$VIN’) and push **Search** button,
- If ‘\$VIN’ is not found, raise an error message “VIN is not found”.
- If ‘\$VIN’ is for an unsold Vehicle, raise message “Unsold Vehicle”
- Else, Display VIN, Vehicle type, Model Year, model name, manufacturer, and colors of the sold **Vehicles**.
- Push **Search Again** to go back to search

(Subtask: **view open repair event**)

- Find the Open **Repair Event** associated with the Vehicle
- Find the **Needs** and **Parts** associated with Repair Event
- Display the labor cost, total cost, odometer, start date, parts number, parts quantity

(Subtask: **add a repair event**)

- If no open Repair Event exists, the user push **Add New Repair**;

- Push a *Customer* button to the Search/Add Customer form to find or add *Customer*
- The user enters the *odometer*
- User push **Add Repair**:
  - If the information is not completed or invalid:
    - Raise a message “Invalid or incomplete information”
  - Else,
    - Add a new *Repair Event* instance entered information;
    - Save the status of *Repair Event* is “Open”;
    - Save the current date as the start date;
- Push **Continue** Go to **update repair event**
- Push **Cancel**, No Repair Event added, display the **Add New Repair** button on the form

(Subtask: **update Repair Event**)

- If there is an open *Repair Event*, Push **Update** button
  - User optionally enter a *description* in the input field
  - User optionally enter a new *labor cost* (“\$*labor cost*”) in an input field
  - Push **OK** button following the input field
    - If “\$*labor cost*” not higher than original and the user is not Owner, or the data is not valid:
      - Raise a message “Invalid entry, Please enter again”
    - Else:
      - Update labor cost, total cost
      - **view open repair event**
  - User optionally enter the *quantity*, *vendor*, *part number*, *price* of *Parts* in an input field
  - Push **OK** button following the input field
    - If the information is valid:
      - Update information
      - **view open repair event**
    - Else:
      - raise message “Invalid or incomplete information”
  - Push **Save for Later**
    - Update information of the *Repair Event* and the *Needs*
    - Status of the *Repair Event* is “Open”.
    - **view open repair event**
  - Push **Save and Close Repair**
    - Update the *Repair Event* and *Needs*
    - Status of *Repair Event* is “Close”
    - Save the current date of *Repair Event* as end date
    - Close the form
  - Push **Cancel**:
    - No Information updated

## View Sales by Color Report

**Lock Types:** Read lock for *Vehicles* and *Sales Events*.

**Number of Locks:** 2

**Enabling Conditions:** Manager or Owner login and push the button on Main Menu

**Frequency:** low, at least once a month

**Consistency(ACID):** Consistency is not critical

**Subtasks:** Mother task is not needed. Order is not necessary.

### View Sales by Color Report

- Page open by clicking on ***Sales by Color Report*** button from the **Main Menu**
- Get the sold date from **Sale Events**; Get the last sold date in all the **Sales Events**
- Find the sold **Vehicles** of each Sales Event; Get the colors of the **Vehicles**
- Display a table with three columns: “Sales within 30 days”, “Sales within last year”, “All time sales”
  - Aggregate number of sold **Vehicles** by color, where sold date  $\leq$  last sold date - 30, as Last 30 days' sales
  - Aggregate number of sold **Vehicles** by color, where sold date  $\leq$  last sold date - 365, as Last year sales
  - Aggregate number of all sold **Vehicles** by color, as All time sales
  - In all the aggregations, if the vehicle colors have multiple values:
    - Aggregate the number to the row “multiple” separately
  - If no **Vehicle** sold in a color in each aggregation:
    - Assign “0” to the value for that color
- Display the Last 30 days sales in the first column
- Display the Last year sales in the second column
- Display the All time sales color in the third column
- Push **Close** to return to **Main Menu**

### View Sales by Type Report

**Lock Types:** Read lock for **Vehicles** and **Sales Events**.

**Number of Locks:** 2

**Enabling Conditions:** Manager or Owner login and push the button on **Main Menu**

**Frequency:** low, at least once a month

**Consistency(ACID):** Consistency is not critical

**Subtasks:** Mother task is not needed. Order is not necessary.

### View Sales by Type Report

- Page open by pushing ***Sales by Type Report*** button on **Main Menu**
- Get the sold dates from **Sale Events**; Get the last sold date in all the **Sales Events**
- Find the sold **Vehicles** of each Sales Event; Get the type of each Vehicle
- Display a table with three columns: “Sales within 30 days”, “Sales within last year”; “All time sales”
  - Aggregate number of sold **Vehicles** by type, where sold date  $\leq$  last sold date - 30, as Last 30 days' sales
  - Aggregate number of sold **Vehicles** by type, where sold date  $\leq$  last sold date - 365, as Last year sales
  - Aggregate number of all sold **Vehicles** by type, as All time sales
  - If no **Vehicle** sold in a color in each aggregation:

- Assign “0” to the value for that type
- Display the Last 30 days sales in the first column
- Display the Last year sales in the second column
- Display the All time sales color in the third column
- Push *Close* to return to Main Menu

## View Sales by Manufacturer Report

**Lock Types:** Read lock for [Vehicles](#) and [Sales Events](#).

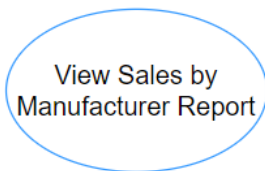
**Number of Locks:** 2

**Enabling Conditions:** Manager or Owner login and push the button on Main Menu

**Frequency:** low, at least once a month

**Consistency(ACID):** Consistency is not critical

**Subtasks:** Mother task is not needed. Order is not critical.



- Page open by pushing *Sales by Manufacturer Report* button on Main Menu
- Get the sold dates from [sale events](#); Get the last sold date in all the [Sales Events](#)
- Find the sold [Vehicles](#) of each Sales Event; Get the manufacturer of each Vehicle
- Display a table with three column: “Sales within 30 days”, “Sales within last year”; “All time sales”
  - Aggregate number of sold [Vehicles](#) by manufacturer, where sold date  $\leq$  last sold date - 30, as Last 30 days sales
  - Aggregate number of sold [Vehicles](#) by manufacturer, where sold date  $\leq$  last sold date - 365, as Last year sales
  - Aggregate number of all sold [Vehicles](#) by manufacturer, as All time Sales
  - If no Vehicle sold in a color in each aggregation:
    - Assign “0” to the value for that type
  - If all three values of a manufacturer is “0”:
    - Remove the row of the manufacturer
- Display the Last 30 days sales in the first column
- Display the Last year sales in the second column
- Display the All time Sales color in the third column
- Push *Close* to return to Main Menu

## View Gross Customer Income Report

**Lock Types:** Read locks for [Customers](#), [Repair Events](#), [Sales Events](#), [Vehicles](#), [Privileged Users\(Salespeople, Service Writer \)](#)

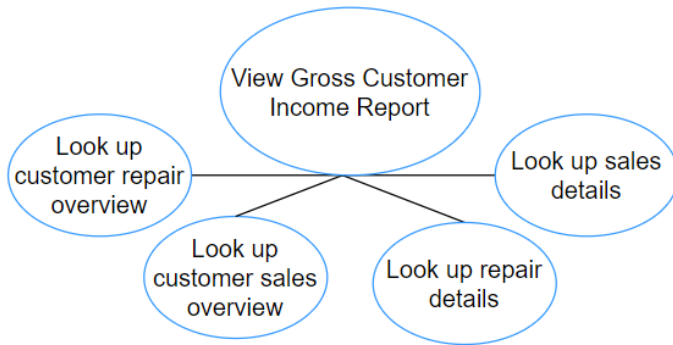
**Number of Locks:** 5

**Enabling Conditions:** Manager or Owner login and pushing the button on Main Menu

**Frequency:** low

**Consistency(ACID):** Consistency is not critical

**Subtasks:** Mother task is required. Order in critical. The Look up customer repair overview and Look up customer sales overviews must be done in parallel, and should go first. The look up repair details, look up sales details should be done together in parallel, and they will not be done until pushing *More Detail*.



- Page open by pushing **Gross Customer Income Report** button on **Main Menu**
- Get the name of **Customers**:
  - If **Person**: get first name + last name
  - Else: get **Business** name

(Subtask: **look up customer sales overview**)

- Find the **Sales Events** associated with customers; Get the sale price, sold date of the **Sales Events**
  - Sum the sale price by customer as customer sales income
  - Aggregate the number of sales by customer as customer sales number
  - Get the max and min of repair start date of customers

(Subtask: **look up customer repair overview**)

- the **Repair Events** associated with customers; Get the repair cost, start date of the **Repair Events**
  - Sum the repair cost by customer as customer repair income
  - Aggregate the number of repairs by customer as customer repair number
  - Get the max and min of repair start date of customers

- Get the total gross incomes by customer repair income + customer sales income
- Get the date of first sale/repair by min(first sold date, first repair date)
- Get the date of last sale/repair by max(last sold date, last repair date)
- Sort the result by total gross incomes descending and by date of last sale/repair descending
- Display the first 15 result of customers' name, date of first sale/repair, date of last sale/repair, total gross incomes in rows

- Push the **More Detail** button of a customer to open the drill-down
- Get the name of the selected customer

(Subtask: **Look up customer sales details**)

- Find **Sales Events** associated with the customer; Get sold date, sale price of **Sales Events**
- Find **Vehicles** of the Sales Events: Get VIN, model year, manufacturer, model name,
- Find **Salesperson** of the Sales Events: Get Salespeople's name
- Display the sale date, sale price, VIN, model year, manufacturer, model name, and Salesperson's name in rows
  - Sort by sale date descending and VIN ascending

(Subtask: **Look up customer repair details**)

- Find **Repair Events** of the customer; get start day, end day, odometer, part cost, labor cost, total cost
- Find the **Vehicles** associated with the Repair Events: get the VIN
- Find the **Service Writer** associated with the Repair Event: Get the Service Writer's name
- Display the start date, end date, VIN, odometer, part costs, labor cost, total cost Service Writer's name
  - Sort the result in sale start date descending, end date descending, VIN ascending

- List incomplete [Repair Events](#) first
- If end date is Null: do not show the end date

- Push **Cancel** to close the drill-down
- Push **Close** to return to Main Menu

## View Repairs by Manufacturer/Type/Model Report

**Lock Types:** Read locks for [Repair Events](#), [Vehicles](#), [Needs](#), [Parts](#)

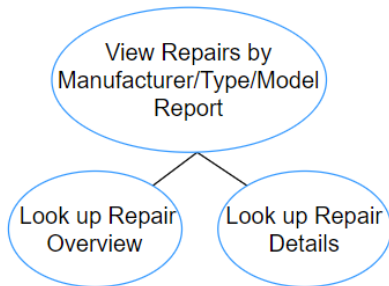
**Number of Locks:** 4

**Enabling Conditions:** Manager or Owner login and push the button on Main Menu

**Frequency:** low

**Consistency(ACID):** Consistency is not critical

**Subtasks:** Mother task is required to coordinate subtasks. Order is critical. The Look up Repair Overview subtask must be done and should go first. The Look up Repair Details subtask should go next and will not be done until pushing **More Detail**.



- Page open by pushing **Repairs by Manufacturer/Type/Model Report** button on Main Menu

(Subtask: **Look up Repair Overview**)

- Get the manufacturer, types, models of [Vehicles](#)
- Find [Repair Events](#) for the Vehicles; Get the total cost, labor cost, part cost in [Repair Events](#)
  - Aggregate the count of repairs, total cost, labor cost, and part cost by Vehicle
  - Aggregate the results of [Vehicle](#) by manufacturer as manufacturer repair count, manufacturer total cost, manufacturer labor cost, manufacturer part cost
  - Sort the result by manufacturer ascending
  - Display manufacturer repair count, manufacturer total cost, manufacturer labor cost, manufacturer part cost
- Include manufacturers if do not have record

(Subtask: **Look up Repair Details**)

- Push the **More Detail** button of each row to open drill down
- Find the [Vehicles](#) in the in manufacturer selected; Get the types and models of the [Vehicles](#)
  - Aggregate the results of [Vehicle](#) by type as type repair count, type total cost, type labor cost, type part cost
  - Sort the result by type repair count descending
    - Aggregate the results of [Vehicle](#) by model as model repair count, model total cost, model labor cost, model part cost
    - Sort the result by model repair count descending of each type
  - Display type repair count, type total cost, type labor cost, type part cost in rows
  - Exclude types if do not have record



- After each type row, display model repair count, model total cost, model labor cost, model part cost
- Exclude types if do not have record
- Push **Cancel** to close the drill-down
- Push **Close** to return to Main Menu

## View Below Cost Sales Report

**Lock Types:** Read locks for [Vehicles](#), [Sales Event](#), [Customers](#), [Salespeople](#)

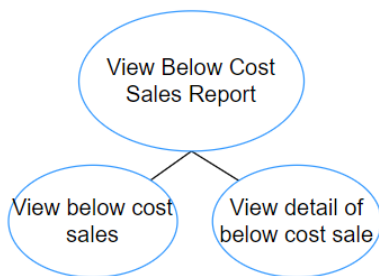
**Number of Locks:** 4

**Enabling Conditions:** Manager Owner login and push the button on Main Menu

**Frequency:** low

**Consistency(ACID):** Consistency is not critical.

**Subtasks:** Mother task is required to coordinate. Order is critical. View Below Cost Sales must be done, and must go first. The View Detail subtask will go next, and will not be done until pushing **More Detail**.



- Page open by pushing **Below Cost Sales Report** button on Main Menu

(Subtask: **view below cost sales**)

- Get sale price and sale date in [Sales Events](#)
- Find [Vehicles](#) associated with Sales Events; Get invoice price of [Vehicles](#)
- Select the Vehicles sold below cost by where(Vehicle.invoice price > Sales Event.sale price);
  - Get the price ratio by sale price/invoice price

(Subtask: **view detail of below cost sale**)

- Find [Customer](#) for the selected the Vehicle
- If the customer is a [person](#):
  - get first name and last name
- Else:
  - get business name
- Find [Salesperson](#) for the selected Vehicles; Get the name of the [Salesperson](#)
- Sort the result by sale date descending and price ratio descending
- Display sale date, sale price, price ratio, customer name, Salesperson name in rows
- If price ratio less than or equal to 95%:
  - Highlight background in red

## View Average Time in Inventory Report

**Lock Types:** Read locks for [Vehicles](#) and [Sales Event](#)

**Number of Locks:** 2

**Enabling Conditions:** Manager Owner login and push the button on Main Menu

**Frequency:** low

**Consistency(ACID):** Consistency is not critical. Order is not necessary

**Subtasks:** Mother task is not required.

### View Average Time in Inventory Report

- Page open by pushing **View Average Time in Inventory Report** button on **Main Menu**
- Get the sold date in **Sales Events**
- Find **Vehicles** associated with Sales Events; Get the date added and type of **Vehicles**
  - Get the inventory time of each sold **Vehicle** by sold date - date added
  - Average the inventory time by type, as type average inventory time
- Display type, type average inventory time in rows
- If no sales history:
  - Display type: "N/A"
- Push **Close** to return to **Main Menu**

## View Part Statistics Report

**Lock Types:** Read locks for **Parts** and **Needs**

**Number of Locks:** 2

**Enabling Conditions:** Manager or Owner login and push the button on **Main Menu**

**Frequency:** low

**Consistency(ACID):** Consistency is not critical. Order is not necessary

**Subtasks:** Mother task is not required.

### View Parts Statistics Report

- Page open by pushing **Part Statistics Report** button on **Main Menu**
- Get the part price, vendor name in **Parts**
- Find the **Needs** for the Parts; Get the quantity used in **Needs**
  - Aggregate quantity used of each **Part** as part total quantity
  - Get the total cost of each part by part price \* part total quantity
  - Sum the part total quantity by vendor name, as vendor total part quantity
  - Sum the total cost of each part by part by vendor name, as total cost of vendor
- Display the vendor names, vendor total part quantity, and total cost of vendor
- Push **Close** to return to **Main Menu**

## View Monthly Sales Report

**Lock Types:** Read locks for **Sales Events**, **Vehicles**, **Salespeople**

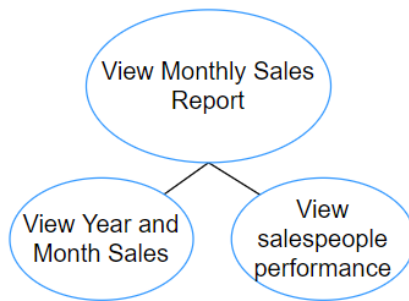
**Number of Locks:** 3

**Enabling Conditions:** Manager or Owner login and push the button on **Main Menu**

**Frequency:** medium, the most frequent viewed report

**Consistency(ACID):** Consistency is not critical

**Subtasks:** Mother task is required to coordinate subtasks. The View Year and Month Sales must be done. Order is critical. The View Salespeople Performance should go next, and it will not be done until pushing **More Detail**.



- Page open by pushing **Monthly Sales Report** button on **Main Menu**

(Subtask: **view year and month sales**)

- Get the sale date, sale price in **Sales Events**
- Find the **Vehicle** associated with Sales Events; get the invoice price in **Vehicle**
  - Aggregate number of **Sales Events** by month and by year, as month/year sold vehicle count
  - Sum the invoice price by month/year as month/year invoice price
  - Sum the sale price by month/year as month/year sale income
  - Get the month/year net income by month/year sale price – month/year invoice price
  - Get the month/year ratio by month/year sale price / month/year invoice price
- If no sales in year/month:
  - Exclude the year/month
- Sort the result by year and month descending
- Display year, month, month/year sold vehicle count, month/year sale income, month/year net income or month/year ratio
  - If month/year ratio greater than or equal to 125%:
    - Highlight row in green background
  - If month/ratio less than or equal to 110%:
    - Highlight row in yellow background

(Subtask: **View salespeople performance**)

- Push the **More Detail** button of each month/year to open drill down
- Find the **Salespersons** of Sales Events; get the name of **Salespersons**
  - Aggregate the number of sales of each **Salesperson** by month/year as salesperson month/year sales count
  - Sum the sale price of **Salespersons** income by salesperson month/year income
- Sort the salesperson month/year sales count descending and by salesperson month/year income descending
  - Display the Salesperson's name, salesperson month/year sales count, salesperson month/year income
  - Push **Cancel** to close the drill-down
  - Push **Close** to return to **Main Menu**