2022-2023

**Group Members**

Ramya Panchatcharam- u2083209

Anil Reddy Seelam - u2197378

Seetharam Murakonda- 2114565

Rashmi katuwalparajuli - u2082353

***Seetharam Murakonda***

u2114565

***Rashmi katuwalparajuli***

u2082353

Vehicle Mart Database

Design and Implementation

**CN5000 Database system**

University of east london

Contents

[1. Introduction 3](#_Toc121995040)

[1.1 Objectives 3](#_Toc121995041)

[2. Abstract 3](#_Toc121995042)

[3. project justification 4](#_Toc121995043)

[4. Functional specification 4](#_Toc121995044)

[5. Entities and ERD in Vehicle Mart Database 4](#_Toc121995045)

[5.1 Entity Relation Diagram (ERD) 4](#_Toc121995046)

[5.1.1 The scenario of the use of ERD for Vehicle Mart 5](#_Toc121995047)

[6. Use cases for Vehicle Mart 7](#_Toc121995048)

[6.1 Use cases Description 8](#_Toc121995049)

[7. Data Dictionary for All the Main Entities 9](#_Toc121995050)

[8. Creation of tables in SQL with Normalization using Oracle 12](#_Toc121995051)

[8.1 Branch Table 12](#_Toc121995052)

[8.2 Vehicle Table 13](#_Toc121995053)

[8.3 Customers table 15](#_Toc121995057)

[8.4 Agent table 17](#_Toc121995061)

[8.5 Dept Table 19](#_Toc121995064)

[8.6 Employees Table 20](#_Toc121995068)

[8.7 Booing Table 22](#_Toc121995071)

[8.8 Payment Table 25](#_Toc121995075)

[8.9 DeletedCustomers table 26](#_Toc121995076)

[9. SQL queries through Oracle Database 27](#_Toc121995077)

[9.1 Show the current price of Sedan in all the Cities 27](#_Toc121995078)

[9.2 Show the largest of price value from all models associated with London and Manchester 27](#_Toc121995079)

[9.3 Show the minimum price for London vehicles (from all models) 29](#_Toc121995080)

[9.4 Find an Agent who has the maximum number of timeslots 30](#_Toc121995081)

[9.5 Find the customer which booked the maximum number of timeslots 31](#_Toc121995082)

[9.6 Show all models having prices higher than average for the model in Birmingham 32](#_Toc121995083)

[9.7 Update the price for all models, for London and Manchester, for today, assuming they want to give an offer of 80 GBP per vehicle. 34](#_Toc121995084)

[9.8 Show the maximum price sold from all Gold and Bronze Agents, for all models 36](#_Toc121995085)

[9.9 Create a trigger which places the customers, which have been deleted into a customer table. This allows records to be maintained while improving query times for existing customers. 41](#_Toc121995089)

[10. Report Reflection 43](#_Toc121995090)

[11. Gantt chart 43](#_Toc121995091)

[12. Minutes of the meeting 44](#_Toc121995092)

[13. Evaluation of the coursework 45](#_Toc121995093)

[14. Conclusion 45](#_Toc121995094)

[References 46](#_Toc121995095)

[Appendix 46](#_Toc121995096)

# Table Description automatically generated

# **1. Introduction**

The main motivation of this project is to alter the old-fashioned management system into an advanced database management system to save time money and resources. And can monitor the overall system from one place. Our group completed project work on designing, developing, and documenting a prototype database system of one of the vehicle sale and rental companies. This company the “Vehicle Mart” is based in London, Manchester, and Birmingham, and expanding to other cities soon. A Vehicle Mart company specializing, in the sale and rental of vehicles to customers. It was a simple and small company where the customer needs to go in person to the office for any small action/service. After this new database management system, the customer can reserve their time slots and vehicle online, via phone, and in person. The CEO governed all branches from the head office. This final coursework report provides a detailed picture of vehicle mart database design and implementation.

## **1.1 Objectives**

* The customers can book their timeslots and vehicles either online/by phone or in person.
* The system has their agents’ and dealers’ details with their level (gold, silver, bronze).
* The CEO from Head Office can handle and manages the overall profit of the agency.
* The security protocols (antivirus and firewall) and prediction modeling will be in place in the future.
* Each branch has its own CTO to monitor total vehicles sold/rented and loss/gain

# **2. Abstract**

With a view to altering the old file system into a new and advanced database management system, we worked on this project. This Vehicle Mart Company set up a hybrid service system. We clearly show ERD, USE Case, and Data Dictionary through SQL statements in Oracle. The significance of this project is booking options online/phone or in person, agent and details with their level, overall management from one place, and space to put prediction modelling in the future.

# **3. project justification**

|  |  |
| --- | --- |
| **Existing Systems** | **New Systems (Proposed)** |
| * Lots of manual and paperwork. * Customers must have to go to the office for services * Timeslots cannot be booked by the customer * No formal feedback is available. | * Time efficiency * The customer will give the feedback directly in a formal way. * Easy to book their timeslot for either Purchasing or Leasing the Vehicle. * Easily handle and manages from one place (the head office). |

# **4. Functional specification**

The system has the Agent and their details including their level (gold, silver, and bronze). They required their login details to gain access to the system. These Agents can give a quote and customers have the right to accept/reject it. The ERD and USE Case, Data Dictionary, and SQL statements in Oracle or Other SQL have been used.

# **5. Entities and ERD in Vehicle Mart Database**

Major entities we find Vehicle Mart database are,

* 1. Branch
  2. Vehicle
  3. Customers
  4. Agent
  5. Booking
  6. Employees
  7. Dept
  8. Payment
  9. DeletedCusotmers

## **5.1 Entity Relation Diagram (ERD)**

Entity-Relationship Model (ERD) defines the structure representation of database, In other words, ERD is a type of flow chart that explains entities, attributes, and their relationship of each other within a database system and ERD is a term which identifies, clarifies, and organise the requirements of the system or database.Using ERD, we can easily implement our database.

Through Entity Relationship diagram we can find which entity are strong and week. In the particular entities, we can easily find attributes was primary key (unique and not null), foreign key(Primary key in other table, null and duplicate allowed, it is a reference) and other constraint such as check, default etc.,

### **5.1.1 The scenario of the use of ERD for Vehicle Mart**

Er diagram for Vehicle mart is created and developed based upon the requirements of the company.

The main purpose of the ERD diagram to use in the vehicle mart is to help the customer in booking and to aid the customer either in the rental/sales or to help the customer with every detail or support from different administrative.

This ERD graphic explains Vehicle Mart Agency, a company that sells and rents vehicles. The vehicle rental and sales system are modelled in this entity relationship diagram. The Vehicle Mart Agency's entity-relationship diagram displays all the visual components of database tables as well as the relationships between renting or selling a vehicle. It established a connection among structured data groups of Vehicle rental system functionalities using structured data. Vehicles, employees, customers, branches, booking, payments, etc. are the major components of the vehicle sale and rental firm.

**Diagram, schematic

Description automatically generated**

# **6. Use cases for Vehicle Mart**

Diagram

Description automatically generated

The use case diagram is a graphic depiction of the interactions among the elements of the vehicle mart. It represents the term to analysis to identify, clarify, and organize the system requirements of the vehicle mart system. The purpose of the use case diagram in vehicle mart is to show how different actors interact to accomplish the goal which is booking a vehicle either for renting or owning. The main actors from the use case diagram are the customer, CTO, receptionist, CEO, and agent. So, the USE case diagram represents the general use case of a vehicle mart booking system. It shows the general processes or functions of the system. That is based on the activities done by the users and actors during reservations. And the broken arrows are indications that the following diagrams connected to them are parts of a process that could either be one of the indications which were included or extended.

## **6.1 Use cases Description**

|  |  |
| --- | --- |
| **Use case name** | **Vehicle Sales and Rental Use case** |
| Actor | Customer, Agency, Agent, CTO, CEO |
| Purpose | * Customer manages their own details, transaction records * Agency manages the customer info and status & info of the vehicle * The agent takes bookings and helps customers with their inquiries and transactions. * CTO manages the monthly reports of the sales/rental and loss/gain of the company * CEO manages the overall reports and the business of the company |
| Pre-condition | * The customer updates the customer details about the booking of the car rental system. * Agency manages and helps the customer throughout the booking. * Agent will take the customer’s login details in order to help in the queries. * CEO will investigate the reports of the company. |
| Post-condition | * The customer must select the type of booking according to their need. * Agency will save the details of each customer details and total bookings. * Agent will update the car details and description to the agency. * CTO will examine the total number of Sales/Rentals during every month. * CEO will examine the overall profit of the company from all agencies. |
| Assumptions | * Agency will assign an agent to the customer. * The agent is connected to the customer regarding the details of the car. * CTO investigates the total vehicles sold/rented in every month. * CEO will manage and develops the company. |

# **7. Data Dictionary for All the Main Entities**

A data dictionary contains data about the database(metadata). It is very important, and it contains details like what is a database table, who is allowed to accessible, and where is the database table stored physically etc., (Meador, 2020)

1. *Data Dictionary for Branch details*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Field Name | Data type | Field Size | Constraints | Description | Example |
| BranchID | Integer | 6 | Primary Key | Unique id for each branch | 1001 |
| Address | Varchar2 | 100 | Null | Address of branch | 28\_Stirling\_RD\_E139NH |
| City | Varchar2 | 25 | Not Null | City of branch | London |
| Phone | Number | 15 | Null | Phone number of branch | 02045369526 |

1. *Data Dictionary for Vehicle information*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Field Name | Data type | Field Size | Constraints | Description | Example |
| VehicleID | Integer | 6 | Primary key | Unique id for each vehicle | 2401 |
| VehicleName | Varchar2 | 25 | Not Null | Name of the vehicle | BMW |
| VehicleType | Varchar2 | 15 | Not Null | Type of Vehicle | *COUPE* |
| Model | Varchar2 | 25 | Not Null | Model of vehicle | M8 |
| Year | Number | 4 | Not Null | Year of vehicle | 2020 |
| Purchase Date | Date | 12 | Null | Purchase date of vehicle | *18-JAN-2020* |
| RentalPrice | Number | 6 | Not Null | Rental price of the vehicle per week | 3800 |
| SalesPrice | Number | 6 | Not Null | Rental price of the vehicle per week | 70000 |
| BranchID | Integer | 6 | Null and Foreign key | Branch id of the vehicle | 1001 |

1. *Data Dictionary for Agent information*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Field Name | Data type | Field Size | Constraints | Description | Example |
| AgentID | Integer | 6 | Primary Key | Unique id for each agent | 1125 |
| AgentName | Varchar2 | 25 | Not Null | Name of agent | Kannika kalai |
| Levels | Varchar2 | 10 | Null | Level of agent. Initial value null | Gold/ Silver/Bronze |
| HireDate | Date | 12 | Not Null | Join date of agent | 01-Jan-2010 |
| Phone | Number | 15 | Not Null | Phone number of agent | 07448246785 |
| Email | Varchar2 | 15 | Default and check | Email id of agent | samram@gmail.com |
| Password | Varchar2 | 15 | Not Null | Password of agent | manvit8@ |
| BranchID | Integer | 6 | Null and Foreign key | Branch id of agent | 1001 |

1. *Data Dictionary for Customer information*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Field Name | Data type | Field Size | Constraints | Description | Example |
| CustomerID | Integer | 6 | Primary Key | Unique id for each customer | 1213 |
| FirstName | Varchar2 | 10 | Not Null | First name of customer | Anil |
| LastName | Varchar2 | 10 | Null | Last name of customer | Reddy |
| Address | Varchar2 | 100 | Not Null | Address of customer | 11 new RD E136GP |
| City | Varchar2 | 15 | Not Null | City of customer | London |
| Phone | Number | 15 | Null | Phone number of customer | 07448246785 |
| Email | Varchar2 | 15 | Default and check | Email id of customer | anil@gmail.com |
| Password | Varchar2 | 15 | Not Null | Password of customer | Ani123d4@ |

1. *Data Dictionary for Dept Details*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Field Name | Data type | Field Size | Constraints | Description | Example |
| DeptID | Integer | 6 | Primary Key | Unique id for each dept | 10 |
| Dname | Varchar2 | 35 | Not Null | Name of the dept | Accountant |
| Location | Varchar2 | 25 | Null | Location of the dept | London |
| Phone | Number | 15 | Null | Phone number of customer | *0203400123* |

1. *Data Dictionary for Employee Details*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Field Name | Data type | Field Size | Constraints | Description | Example |
| EmpID | Integer | 6 | Primary Key | Unique id for each employee | 20809 |
| FirstName | Varchar2 | 25 | Not Null | First name of employee | Vetrivel |
| LastName | Varchar2 | 25 | Null | Last name of employee | Panchatcharam |
| Role | Varchar2 | 25 | Not Null | Role of employee | Receptionist |
| HireDate | Date | 12 | Not Null | Join date of agent | '01-jan-2006 |
| Salary | Decimal | 7,2 | Null | Salary of employee |  |
| Email | Varchar2 | 25 | Default and check | Email id of employee | vetri1976@gmail.com |
| BranchID | Integer | 6 | Null and Foreign key | Branch id of employee | 1001 |
| DeptID | Integer | 6 | Null and Foreign key | Dept id of employee | 20 |

1. *Data Dictionary for Booking Details*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Field Name | Data type | Field Size | Constraints | Description | Example |
| BookingID | Integer | 6 | Primary Key | Unique id for each booking | 1 |
| BookingType | Varchar2 | 25 | Not Null | First name of employee | sale |
| BookingDate | Date | 12 | Null and Check | Date of booking | 15-Nov-2022 |
| DateOut | Date | 12 | Null and Check | Date of vehicle out from agency | 25-Nov-2022 |
| DateBack | Date | 12 | Null and Check | It applicable only for rental not for sale. Rental car come back vehicle to agency | Null |
| CustomerID | Integer | 6 | Null and Foreign key | Customer id of booking | 1215 |
| EmpID | Integer | 6 | Null and Foreign key | Employee id of booking | 20809 |
| VehicleID | Integer | 6 | Null and Foreign key | Vehicle id of booking | 2401 |
| AgentID | Integer | 6 | Null and Foreign key | Agent id of booking | 1114 |

1. *Data Dictionary for Booking Details*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Field Name | Data type | Field Size | Constraints | Description | Example |
| PaymentID | Integer | 6 | Primary Key | Unique id for each payment | 1 |
| PaymentDate | Date | 12 | Null | Date of payment | 15-Nov-2022 |
| Amount | Number | 7 | Not Null | Amount for booking | 70000 |
| AddCharge | Decimal | 7 | Null | Additional charge for booking | 200 |
| BookingID | Integer | 6 | Null and Foreign key | Customer id of booking | 1 |

1. *Data Dictionary for DeletedCustomers information*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Field Name | Data type | Field Size | Constraints | Description | Example |
| DC\_ID | Integer | 6 | Primary Key | Unique id for each deleted customer and we do through sequence | 1 |
| FirstName | Varchar2 | 10 | Not Null | First name of customer | Anil |
| LastName | Varchar2 | 10 | Null | Last name of customer | Reddy |
| Address | Varchar2 | 100 | Not Null | Address of customer | 11 new RD E136GP |
| City | Varchar2 | 15 | Not Null | City of customer | London |
| Phone | Number | 15 | Null | Phone number of customer | 07448246785 |
| Email | Varchar2 | 15 | Default and check | Email id of customer | anil@gmail.com |
|  |  |  |  |  |  |
| CustomerID | Integer | 6 | Foreign Key | Unique id for each agent | 1213 |

# 8. Creation of tables in SQL with Normalization using Oracle

There are eight tables created for Vehicle Mart with normalization. Such as

1. Branch Table
2. Vehicle Table
3. Agent Table
4. Customers Table
5. Booking Table
6. Employees Table
7. Dept Table and
8. Payment Table
9. DeletedCustomers Table

All the table’s detailed descriptions (like all the entities, their attributes, primary key, foreign key, default, and check constraints) with proof are given below:

## **8.1 Branch Table**

* *Entity: Branch*
* *Attributes: BranchID, Address, City, Phone*
* *Primary key: BranchID*
* *Relationship: Branch to Employee, Agent, and Vehicle(1:M)*

The Branch table is a must for Vehicle Mart because it has more than one branch and branch details can be stored in the branch database table. The Branch table has the branch id, address, city, and phone. Here BranchID is the primary Key constraint and the branch table link to Vehicle, Agent, and Employees tables.

*Table creation and description for branch table:*

Graphical user interface, application

Description automatically generated

*Data Insertion and selection in Branch table:*

*Graphical user interface, text, application

Description automatically generated*

## **8.2 Vehicle Table**

* *Entity: Vehicle*
* *Attributes: VehicleID, VehicleName, VehicleType, Mode, Year, PurchaseDate, RentalPrice, SalesPrice, and BrachID)*
* *Primary key: VehicleID*
* *Foreign Key: BranchID*
* *Check Constraint: SalesPrice and RentalPrice*
* *Relationship: Vehicle to Booking(1:M)*

The Vehicle information is recorded in the vehicle database table and their attributes like a model, year, brand, purchase date, rental price, sales price, and branch id. In this table, the vehicle is the primary key constraint and branch id is the foreign key constraint which means the Vehicle is connected from the branch table through the foreign key (BranchID).

In this table, we use the Check constraint for rentals and sales prices because it must be between 10 to 100,000.

### Vehicle table creation

Graphical user interface, text, application

Description automatically generated

### Vehicle table Insertion:

Text

Description automatically generated

### Vehicle table selection:

Graphical user interface, application

Description automatically generated with medium confidence

## **8.3 Customers table**

* *Entity: Customers*
* *Attributes: CustomerID, FirstName, LastName, Address, City, Phone, Email, and Password*
* *Primary key: CustomerID*
* *Default Constraints: Email*
* *Check Constraints: Email*
* *Relationship: Customer to Booking(1:M)*

The Customer database table stores the information of the customer such as customer id, first name, last name, address, city, phone, email, and password. In this table, CustomerID is the Primary key constraint. Customer table link to booking table.

### Customer table creation

Graphical user interface, text, application

Description automatically generated

### Customer table insertion:

Chart, scatter chart

Description automatically generated

### Customer table selection

Graphical user interface, application

Description automatically generated

## **8.4 Agent table**

* *Entity: Agent*
* *Attributes: AgentID, AgentName, HireDate, Phone, Levels, Email, Password, and BranchID*
* *Primary key: AgentID*
* *Foreign Key: BranchID*
* *Default Constraints: Email*
* *Check Constraints: Email*
* *Relationship: Agent to Booking(1:M) and Agent to Vehicle(M:1)*

All the Vehicle Mart agent details are recorded in the agent table. Information such as agent id, agent name, hire date, levels, phone, and branch id. In this table, Agent id is the primary key constraint. Agent link with booking table through foreign keys BranchID.

### Agent table creation

Graphical user interface, text, application

Description automatically generated

### Agent table insertion and selection

Table

Description automatically generated

We use check and default constraints for the email attribute. If the user must use this format when the user enters the email ‘%@%’ and if they not entering the email default email as ‘unknow@gmail.com’.

**Insert All the attributes**

INSERT INTO AGENT VALUES (1114, 'Vishnu Vetri', 'Gold', '11-May-2014', 07448231489, 1001, 'vishnuvetri97@gmail.com', 'vishun@76');

**Insert partially attribute**

INSERT INTO AGENT (AgentId, AgentName, Levels, HireDate, Phone, BranchID, Password) VALUES (1125, 'Kannika kalai', 'Silver', '01-Jan-2010', 07448246785, 1002, 'manvit8@');

## **8.5 Dept Table**

* *Entity: Dept*
* *Attributes: DeptID, DeptName, Location, and Phone*
* *Primary key: DeptID*
* *Relationship: Dept to Employee(1:M)*

The Vehicle Mart department information is stored in the dept database table. Dept table attributes are dept id, dept name, and location. This table is connected to the Employee table. In this table, the department id is uniquely identified through Primary Key (DeptID).

### Dept table creation

Graphical user interface, text

Description automatically generated with medium confidence

### Dept table insertion and selection

### Graphical user interface, text, application Description automatically generated

## **8.6 Employees Table**

* ***Entity:*** *Employees*
* ***Attributes:*** *EmpID, FirstName, Lastname, Role, HireDate, Salary, Email, BranchID, and DeptID)*
* ***Primary key:*** *EmpID*
* ***Foreign Key:*** *DeptID*
* ***Default Constraints:*** *Email*
* ***Check Constraints:*** *Email*
* ***Relationship:*** *Employee to Booking(1:M) and Employee to Branch(M:1)*

The Employees database table stores the details of the employees who are working in the Vehicle Mart Company. The information such as employee id, first name, last name, role, hire date, salary, email, branch id, and dept id. Employees (Receptionists) can manage the customer information in the database system. branch and dept tables are connected to the employee table through a foreign key (BranchID, DeptID). We set the foreign key through alter command for all the foreign keys and here as well check the email like customer table.

### Employees table creation

Graphical user interface, text, application

Description automatically generated

### Employee table insertion and selection

Graphical user interface, application

Description automatically generated

## **8.7 Booing Table**

* *Entity: Booking*
* *Attributes: BookingID, BookingDate, DateOut, DateBack, BookingType, CustomerId, EmployeeId, VehicleId, and AgentId*
* *Primary key: BookingID*
* *Foreign Key: CustomerId, EmployeeId, VehicleId, and AgentId*
* *Relationship: Booking to Payment(1:M) and Booking to Agent,Vehicle,Customer, and Employee(M:1)*

The sold and renal vehicle booking details are stored in the booking table. The information such as booking id, booking date, date out (sale or rental date), date back (if it is rental date back we can mention and sale type we enter null), booking type (Sale or rental), customer id, employee id, vehicle id, and agent id.

In this table, the booking id is a primary key. This table gets the information from customers, employees, agents, and vehicle tables through the foreign key (VehicleID, CustomerID, EmpID, and BranchID). Booking date, date out, and date back after the coursework date we check through the check constraint.

### Booking table creation

Text

Description automatically generated

### Booking table insertion

Text

Description automatically generated

### Booking table selection

Graphical user interface, application, table

Description automatically generated

Table

Description automatically generated

In total, we are book twenty-six bookings from different branches, agents, and vehicle.

## **8.8 Payment Table**

* *Entity: Payment*
* *Attributes: PaymentID, PaymentDate, Amount, AddCharge, and BookingID*
* *Primary key: PaymentID*
* *ForeignKey: BookingID*
* *Relationship: Payment to Booking(M:1)*

All the booking payment details are recorded in the payment table. In this table, PaymentID is a primary key and this table connected from the booking table through the foreign key BookingID.

Payment table creation

Graphical user interface, text

Description automatically generated with medium confidence

Here we insert sample record because using payment table there is no queries in our case study.

Graphical user interface, text

Description automatically generated

## **8.9 DeletedCustomers table**

* *Entity: DeletedCusotmers*
* *Attributes: DC\_ID, CustomerID, FirstName, LastName, Address, City, Phone, Email, and Password*
* *Sequence: DC\_ID*
* *Relationship: Customer to DeletedCustomers(1:1)*

This table is mainly for our existing customers. Whenever the admin deletes the customer that will automatically be updated in this table through the trigger (query no 9). Same attributes as the customer but only DC\_ID is a deleted customer table unique identification using sequence. You understand more about this table in the ninth queries.

Graphical user interface, text, application

Description automatically generated

Selection of Deletedcustomers

A screenshot of a computer

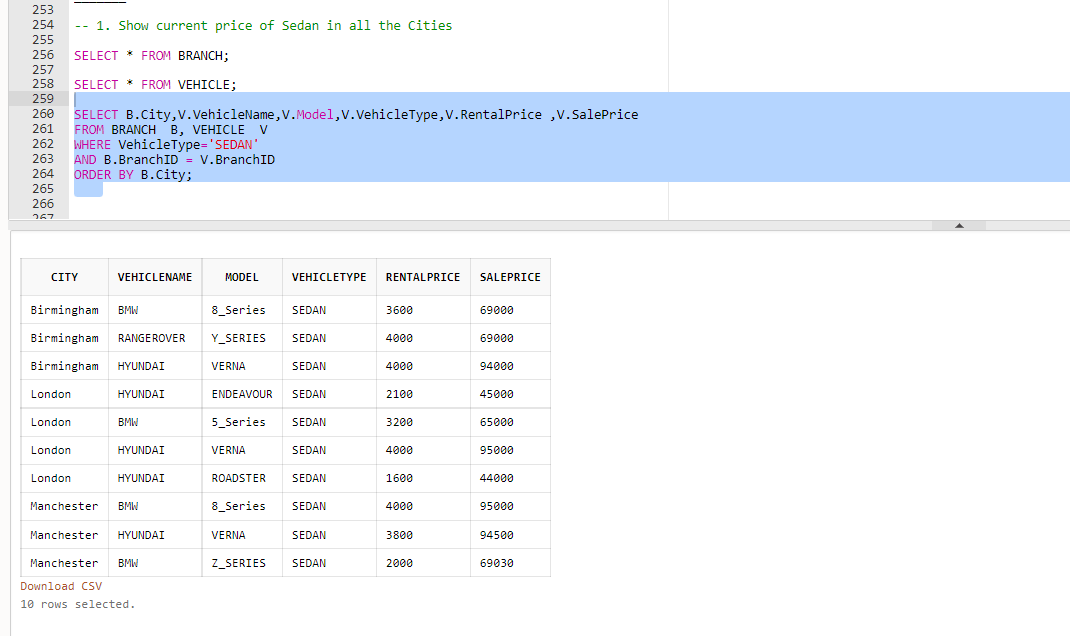
Description automatically generated

# 9. SQL queries through Oracle Database

## **9.1 Show the current price of Sedan in all the Cities**

Using the select command we can display all records and attributes, or we can give the condition and retrieve the partial fields. For this query, we need a branch and vehicle table.

Select the fields: From the branch table select city and vehicle table select vehicle name, model, vehicle type, rental price, and sales price than in the where condition set vehicle type is ‘sedan’, and the vehicle table, branch id and branch table branch must be equal. Finally order by city in ascending order.



## **9.2 Show the largest of price value from all models associated with London and Manchester**

Vehicle and branch tables are important for this query. Select all the records from the vehicle and branch table for our verification. Then do the following,

**Step 1:** Create the *“Price\_AllModel”* view using the branch and vehicle tables. Match the branch id in both tables and select the attributes like a city from the branch table and branch id, sale price, and rental price. The result given below,

Graphical user interface, text, application

Description automatically generated

Graphical user interface, application

Description automatically generated

**Step 2:** Using the *“Price\_AllModel”* view select the attributes like city, vehicle name, model, sales price, and rental price. For the Sales price and rental price, we use the aggregate function max. In the where condition specify city in London and Manchester, group by model, vehicle name, and city, and order by city. The result gives below,

Table

Description automatically generated

## **9.3 Show the minimum price for London vehicles (from all models)**

Using the *“Price\_AllModel”* view select the attributes like city, vehicle name, model, sales price, and rental price. For the Sales price and rental price, we use the aggregate function min. In the where condition specify city in London. Group by model, vehicle name, and city, and order by city. The result gives below,

Table

Description automatically generated with medium confidence

## **9.4 Find an Agent who has the maximum number of timeslots**

To find the maximum number of timeslots by agent mean we need an agent and booking table and we do two steps like create a view to match the agent id and then use that view to find the maximum number of timeslots in the booking.

**Step 1:** Create “AgentView” using select attributes (Agent id and name) only which agent match in both tables then execute AgentView.

Table

Description automatically generated

Table

Description automatically generated

**Step 2:** Using “*AgentView”,* select the attributes such as agent id, agent name, and levels, and use the count aggregate function for the agent id and give the alias name “Number of Booking”. We use the count function must group by agent id, name, and levels.

Order by a “Number of bookings” alias names descending then we get all the agent booking details. We need who to maximum timeslot so fetch first 1 row only give at end means we get the final output.

Graphical user interface, text, application

Description automatically generated

## **9.5 Find the customer which booked the maximum number of timeslots**

Customers and a booking table are needed to find customers who booked the highest number of bookings in the Vehicle mart database system. We just select both tables to get an idea about this query then we do two steps there are,

**Step 1:** Create a view “*CustomerView”* by selecting attributes like booking table customer, customer table first name, last name and then give where condition, in both tables customer id is equal then select the “*CustomerView”* and get the result.

Graphical user interface, application

Description automatically generated

Graphical user interface, application

Description automatically generated

**Step 2:** Using the select query select the attributes like customer id, number of bookings (using count function for customer id), and customer name from the view “*customerview”* then group by customer id and name, order by number of booking descending finally fetch first row from selected query.the

Graphical user interface, text, application

Description automatically generated

## **9.6 Show all models having prices higher than average for the model in Birmingham**

Booking and vehicle tables are important for selecting all the models having prices larger than average for the models in the city of Birmingham. For this, we did three steps. There are,

**Step 1:** Create the view name called *“Vehicleprice”,* from the booking and vehicle tables select the attributes like city, model, sales price, and vehicle type then gave where condition, select the attributes when customer id matched in both tables. When we execute the *“Vehicleprice”,* view the following result we got.

Graphical user interface, text, application

Description automatically generated

**Step 2:** Create the second view name called *“Avgsaleprice*” by selecting the sale price and applying an average function to it from the view *“Vehicleprice”* then gave where condition, the city is Birmingham, group by model, order by average sale price, and fetch first row from it finally we got average price of Birmingham city. Result given below

Graphical user interface, text, application

Description automatically generated

**Step 3:** Select city sale price and model from the “*vehicleprice”* view then give where condition, and sale price higher than the sub query (select all from “*avgsalprice*” view).

Text

Description automatically generated

## **9.7 Update the price for all models, for London and Manchester, for today, assuming they want to give an offer of 80 GBP per vehicle**.

Branch and vehicle tables are needed for updating the price for all models in London and Manchester and giving an offer of 80 pounds per vehicle. For this, we did three steps. There are,

**Step 1:** Create the view name called *“AllVehicleModel”,* from the branch and vehicle tables select the attributes like city, model, sales price, rental price, and vehicle name then give where condition, select the attributes when customer id matched in both tables. We execute the *“AllVehicleModel”* view the following result we got.

Graphical user interface, application

Description automatically generated

Step 2: We update the “AllVehicleModel “view like new rental price is equal to old rental price minus 80 pounds and the where the condition is city in London and Manchester

Graphical user interface, text, application

Description automatically generated

**Step 3:** selecting the *“AllVehicleModel “*view order by City then we got updated table and it will update the original table vehicle as well.

Graphical user interface, text, application

Description automatically generated

**Vehicle table**

Text, application

Description automatically generated with medium confidence

## 9.8 Show the maximum price sold from all Gold and Bronze Agents, for all models

Vehicle, agent, and booking tables are needed for selecting the maximum price sale from all agents in gold and bronze level for all models. For this, we did four following steps.

**Step 1:** Create the view name called *“Agentid\_Matchview”,* from the booking and vehicle tables select the attributes like agent id, vehicle name, type, model, and sale price then give where condition, select the attributes agent id matched in both tables. We execute the *“Agentid\_Matchview”* view the following result we got.

Graphical user interface, application

Description automatically generated

**Step 2:** Create the second view name called *“HighestPrice\_view”* from the *“Agentid\_Matchview” view, S*elect the attributes like agent id, vehicle name, type, model, and maximum sale price (using the max function for sale price) then group by max sale price and order by agent id, vehicle model, type, and name. We execute the *“HighestPrice\_view”* view the following result we got.

Table

Description automatically generated

Step 3: Create the three-view name called *“Updatelevel”* from the *“Agentid\_Matchview” view, S*elect the attributes like agent id, and the number of bookings then group by agent id and order by number of bookings. We execute the *“Updatelevel”* view the following result we got.

Graphical user interface, application

Description automatically generated

### Agent table before update levels

Graphical user interface, application, table, Excel

Description automatically generated

### View “AgentView1” after updating levels through number of booking

Graphical user interface

Description automatically generated with low confidence

### Agent table after updating levels through number of booking

Graphical user interface, text, application, email

Description automatically generated

Step 4: Create a *“finalagentview”* view through *“hightestprice\_view”*, select needed attributes, match the agent id, and ordered by levels. Select the *“finalagentview”* and where conditions, levels in gold and bronze. Finally, we got maximum price sold from all Gold and Bronze Agents, for all models.

Graphical user interface

Description automatically generated with low confidence

Graphical user interface, application

Description automatically generated

## **9.9 Create a trigger which places the customers, which have been deleted into a customer table. This allows records to be maintained while improving query times for existing customers.**

Customer tables are needed for deleted records, and we need one more table to add deleted customers. For this, we did four following steps.

**Step 1:** For this query we first create *“deletedcustomers”* table which is like customer’s table. The structure of the table given below

Graphical user interface, text, application

Description automatically generated

Step 2: Create sequence for DC\_ID because we going to insert the record in to the *“deletedcustomers”* when the trigger is executed. Sequence gives auto increment by next record add into the table and then create trigger as given below

Graphical user interface, text, application

Description automatically generated

We create a trigger after a delete on customers for each row and insert the deleted record into the *“deletedcustomers”* table. Begin the trigger and insert a value into “deletedcustomers”. Dc\_seq. nextval means de\_seq sequence next value for DC\_Id,: old:customerid means last delete customer id, all other attributes also similar to :old:customerid. Finally, execute the trigger.

**Step 3**: Delete any record from the customer table

**Graphical user interface, text, application

Description automatically generated**

Select the *“deletedcustomers”* table and see the result

Graphical user interface, text, application

Description automatically generated

One more record delete from the customer table

Graphical user interface, application

Description automatically generated with medium confidence

Customer table after deleting two records

Graphical user interface, application

Description automatically generated

# 10. Report Reflection

Database designs give the blueprints of how that data is going to be stored in the system and the designing principles defined for a database (how the request is processed and provide a clear idea of the behaviour of any application. The proper database design meets all the requirements of clients or users and the developing, or processing time of an application is reduced if the constraints of database designing are good (JavaTpoint., 2021)

We are design our database through ERD, use cases, and a data dictionary. After this design, we can easily implement it in oracle. Almost nine queries we had done through our database. Every query result was perfect. For example, In the entity relationship diagram we have to include the customer deleted table so effortlessly we did our query(Delete the customer in the customer table it will automatically be saved to other table) using the trigger concept and it work

In our database design, we can use oracle our DBMS tool was an oracle. It is very secure compared to all other tools. If want, we choose the alternative database tool MySQL. It is very user-friendly and mostly like an oracle. In the future we can create the font end design and using this database we can book the vehicles and check how it will work with front end software.

The main software is used in out vehicle mark database are,

* Oracle for creating database and done all the related queries.
* Notepad for backup the queries for later use
* MS Word for creating coursework report
* MS Excel for creating Minutes for Meeting
* MS Project plan 365 for developing for Gantt chart
* Draw.io for developing ERD, Use case Diagram

# 11. Gantt chart

During the project work, our all activities and action has been recorded in the Gantt chart. The Gantt chart itself is self-explanatory which is given below-

We are use Project plan 365 for developing Gantt chart.

Graphical user interface, application, table, Excel

Description automatically generated

# 12. Minutes of the meeting

A picture containing timeline

Description automatically generated

Table

Description automatically generated

# 13. Evaluation of the coursework

**Ramya(u2083209):** I have got a complete idea about different Vehicle Rental systems. And when doing the project, I have got many doubts in the queries which is my part in the project it takes much time for my queries and to identify the errors but finally I was very happy when I have completed all the queries and my part. Already I know SQL but after completed this coursework I am very confidently say I can be able to work with any database. I have learned more about Vehicle Rental systems and found what are the main entities in the vehicle booking system and how they are interrelated. Now I got an idea about how to initiate the project, and what system design needed, (ER, use case Diagram, data dictionary) is important to implement the project. I am proud of myself because I help all the team members when they needed it. For this coursework we worked hard and understand what we want exactly. I think finally we reach our goal.

**Anil**(**u2197378):** As per the given project we have decided to form a group of 4 people and divided the project to everyone which is about vehicle mart. During the start of my part I don’t have an idea about the SQL and even the database but I have made researched vehicle rental systems and I learned the different ways to describe the use case ERD and to implement the Database. I felt very happy in doing this project and after completing this project, I got complete detailed information about the Vehicle Rental and management system. So, I am especially thankful to my professor for giving these types of projects to us and I strongly recommend giving these types of projects to students so that they will be more helpful to each student in building conversation and in improving different skills.

**Seetharam**(**u2114565):** I Am so happy to share my view on the project which is about VEHICLE MART. So, as my part is ERD in the project, I found what are the different entities in the project and I have created an ERD with 8 different tables. Before doing these projects, I didn’t have any idea about the vehicle rental system but because of these projects, I have learned about the different car rental companies which are in the UK. I have improved my communication skills in this group work, and I also made very good research on the Rental System, so I have enjoyed a lot doing this project.

**Rashmi(u2082353):** Initial (design) to completion stage communication among the team was clear and every member of the team knew about their required tasks. I learned and enjoyed a lot of team composition and teamwork throughout the project. We shared our views and thought whenever we have time and space, it was a flexible approach to work. We don’t have specific means of communication, as per need we communicated either by WhatsApp or face-to-face. We try our best to make this project more professional, knowledgeable flexible, and interesting for all members. We learned more than expected during this short span of project time.

# 14. Conclusion

Our assignment deadline is nearer, and we look back to our group work. we realized that we are open and very honest about our assigned tasks. In our database design, we can use oracle our DBMS tool was an oracle. It is very secure compared to all other tools. If want, we choose the alternative database tool MySQL. It is very user-friendly and mostly like an oracle. In the future, we can create the font end design, and using this database we can book the vehicles and check how it will work with front end software.

# References

JavaTpoint. (2021, 02 15). *https://www.javatpoint.com/database-design.* Retrieved from javatpoint: https://www.javatpoint.com

Meador, D. (2020, 06 19). *https://www.tutorialspoint.com/What-is-Data-Dictionary*. Retrieved from tutorialspoint.com: https://www.tutorialspoint.com

# Appendix

All the database queries text file

\*\*\* Thank you \*\*\*