**Project Team 1 - <The Motor Empire>**

**PROJECT PROPOSAL**

**Content, Scope and Objectives**

**Briefly describe the overall context, scope and objectives of the software for which you propose to design and develop a database (*similar to the high level vision statement given to you for the case study we are using for classroom activities, but more concrete in scope*).**

Our group will utilize mySQL to create a data driven relational database storing the necessary information for a car dealership. We will cover the cars stored and the cars sold by the dealership. The database will also store information about the various cars including make, model, year, and price. We will also keep information on customers and sellers such as names, addresses, emails, credit card numbers, etc. The dealership will be in an online only format and employees will be divided into sellers, managers, and suppliers.

**PROJECT ENVIRONMENT**

**Briefly (in a paragraph or so) describe your project environment, including version numbers, etc. if relevant.**

Our team will utilize local environments consisting of the mySQL database management system. We will also be using AWS to store our relational database. We will be creating/utilizing tables and attributes to effectively store and maintain data. We will also be visualizing this data through schemas such as using an ER diagram to model the relationships between entities.

**HIGH LEVEL REQUIREMENTS**

**Initial user roles**

|  |  |
| --- | --- |
| **User Role** | **Description** |
| Employee | Employee responsible for selling cars to customers |
| Customer | Customer is the purchaser of the cars in the dealership |
| Admin | Update and manage the database accordingly |
| Manager | Hire employees for the company, keeps inventory |
| Supplier | Supplying the dealership with cars |
| Inventory | Tracks dealership car inventory |
| Credit Card | Holds credit information used for purchases |

**Initial user story descriptions**

|  |  |
| --- | --- |
| **Story ID** | **Story description** |
| Customer | As a Customer, I want to search for a car so that I can know the type of car that fits my needs. |
| Customer | As a Customer, I want to purchase a car so that I can drive |
| Customer | As a Customer, I want to use my credit card to buy a car |
| Employee | As a Employee, I want to search for cars in the company’s inventory |
| Employee | As a Admin, I want to sell cars to the customers |
| Admin | As a Admin, I want to make accounts for the customers |
| Admin | As an Admin, I want to make accounts for the employees |
| Supplier | As a Supplier, I want to supply the company cars to make a living |
| Manager | As a manager, I want to hire people to make them employees |
| Manager | As a manager, I want to access/update inventory |

**HIGH LEVEL CONCEPTUAL DESIGN**

Entities:

* CustomerAccount
* EmployeeAccount
* AdminAccount
* Supplier
* ManagerAccount
* Inventory
* CreditCard

Relationships:

* Employee <sells> Car
* Employee <searches> Car
* Employee <messages> Customer
* Customer <purchases> Car
* Customer <searches> Car
* Customer <uses> Credit Card
* Admin <creates>  customer accounts
* Admin <creates> employee accounts
* Supplier <restocks> Inventory
* Manager <hires> Employee
* Manager <checks> Inventory

**Sprint 1**

**REQUIREMENTS**

List your updated user stories and any notes you wish to include in decreasing order of priority and highlight the stories chosen for Sprint 1. *There is no need to show your story refinement process - just the list of updated stories suffices.* Use the format shown below.

|  |  |
| --- | --- |
| **Story ID** | **Story description** |
| Employee | As an employee, I want to search and sell cars to customers |
| Customer | As a customer, I want to search and buy cars from Employees |
| Admin | As an Admin, I want to create accounts for customers and employees |
| Manager | As a manager, I will supervise and oversee my employees |
| Supplier | As a supplier, I want to supply the dealership with cars |
| Inventory | Inventory will hold the values of the number in stock and vehicle details |
| Credit Card | Stores credit card information for customers to utilize during transactions |

**CONCEPTUAL DESIGN**

Include your detailed conceptual design here. Use the format shown below.

Entity: **EmployeeAccount**

Attributes:

**-employeeID**

**-managerID**

**-firstName**

**-lastName**

**-email**

**-phoneNum**

**-position**

Entity**:** **CustomerAccount**

Attributes:

**-customerID**

**-firstName**

**-lastName**

**-email**

**-phoneNum**

**-Address**

**-Balance**

**-creditCardNum**

Entity: **Messages**

Attributes:

**-customerID**

**-employeeID**

**-messages**

Entity: **Credit Card**

Attributes:

**-cardNum**

**-cardHolderName**

**-expDate**

**-CVV**

**-billingAddress**

Entity: **Cars**

Attributes:

**-VINnum**

**-employeeID**

**-customerID**

**-manufacturer**

**-model**

**-retailPrice**

**-numInStock**

Entity: **Supplier**

Attributes:

**-supplierID**

**-name**

**-phoneNum**

**-email**

Entity: **ManagerAccount**

Attributes:

**-managerID**

**-firstName**

**-lastName**

**-phoneNum**

**-email**

**Relationship:** Customer Buy Cars

**Cardinality:** One to Many

**Participation:**

     Customers has total participation

     Cars has partial participation

**Relationship:** Customers Use Credit Card

**Cardinality:** One to Many

**Participation:**

     Customers has total participation

     Credit Card has total participation

**Relationship:** Employees messages Customers

**Cardinality:** Many to Many

**Participation:**

     Employees has total participation

     Customers has total participation

**Relationship:** Manager oversees Employees

**Cardinality:** one to Many

**Participation:**

     Manager has total participation

     Employees has partial participation

**LOGICAL DESIGN**

Include your logical design here. Use the format shown below.

Table: **EmployeeAccount**

Columns:

**-employeeID [Primary Key]**

*-managerID [Foreign Key - REFERENCES ManagerAccount(managerID)]*

-firstName

-lastName

-email

-phoneNum

-position

Table: **CustomerAccount**

Columns:

**-customerID [Primary Key]**

-firstName

-lastName

-email

-phoneNum

-Address

-Balance

*-creditCardNum [Foreign Key - REFERENCES CreditCard(cardNum)]*

Table: **Messages**

Columns:

***-customerID [Foreign Key - REFERENCES CustomerAccount(customerID)]***

***-employeeID [Foreign Key - REFERENCES EmployeeAccount(employeeID)]***

-messages

**[Primary Key is composite key (customerID, employeeID)]**

     Table: **CreditCard**

Columns:

**-cardNum [Primary Key]**

-cardHolderName

-expDate

-CVV

-billingAddress

     Table: **Cars**

Columns:

**-VINnum [Primary Key]**

*-employeeID [FOREIGN KEY (employeeID) REFERENCES EmployeeAccount(employeeID)]*

*-customerID [FOREIGN KEY (customerID) REFERENCES CustomerAccount(customerID)]*

-manufacturer

-model

-retailPrice

-numInStock

     Table: **Supplier**

Columns:

-**supplierID [Primary Key]**

-name

-phoneNum

-email

     Table: **ManagerAccount**

Columns:

-**managerID [Primary Key]**

-firstName

-lastName

-phoneNum

-email

**SQL QUERIES**

List at least **three** SQL queries that perform data retrievals relevant to the features chosen in the current sprint. For each query, paste a **screenshot** of the output, as shown through your user interface.Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

**Graphical user interface

Description automatically generated**

**Sprint 2**

**REQUIREMENTS**

List your updated user stories in decreasing order of priority. Highlight the stories for which database design was completed in Sprint 1 in one color. Highlight the updated/new stories chosen for Sprint 2 in a different color. *There is no need to explicitly show your story refinement process.* Use the format shown below.

|  |  |
| --- | --- |
| **Story ID** | **Story description** |
| Employee | As an employee, I want to search and sell cars to customers |
| Customer | As a customer, I want to search for car details and buy cars from Employees |
| Admin | As an Admin, I want to create and manage accounts for customers and employees |
| Manager | As a manager, I will supervise and oversee my employees and hiring |
| Supplier | As a supplier, I want to supply the dealership with cars and provide number of stock |
| Inventory | Inventory will hold the values for the vehicle details including make, model, VIN |
| Credit Card | Stores credit card information for customers to utilize during transactions |

**CONCEPTUAL DESIGN**

Include your complete updated conceptual design here. Use the format shown below.

Entity: **EmployeeAccount**

Attributes:

**-employeeID**

**-managerID**

**-firstName**

**-lastName**

**-email**

**-phoneNum**

**-position**

Entity**:** **CustomerAccount**

Attributes:

**-customerID**

**-firstName**

**-lastName**

**-email**

**-phoneNum**

**-Address**

**-Balance**

Entity: **EmployeeAccountCustomerAccount**

Attributes:

**-customerID**

**-employeeID**

**-messages**

**-sender**

Entity: **Credit Card**

Attributes:

**-cardNum**

**-cardHolderName**

**-expDate**

**-CVV**

**-customerID**

Entity: **Cars**

Attributes:

**-VINnum**

**-manufacturer**

**-model**

**-year**

Entity: **Supplier**

Attributes:

**-supplierID**

**-name**

**-phoneNum**

**-email**

Entity: **ManagerAccount**

Attributes:

**-managerID**

**-firstName**

**-lastName**

**-phoneNum**

**-email**

Entity: **SearchCars**

Attributes:

**-VINnum**

**-customerID**

**-employeeID**

**Relationship:** Customer Buy Cars

**Cardinality:** One to Many

**Participation:**

     Customers has total participation

     Cars has partial participation

**Relationship:** Customers Use Credit Card

**Cardinality:** One to Many

**Participation:**

     Customers has total participation

     Credit Card has total participation

**Relationship:** Employees messages Customers

**Cardinality:** Many to Many

**Participation:**

     Employees has total participation

     Customers has total participation

**Relationship:** Manager oversees Employees

**Cardinality:** one to Many

**Participation:**

     Manager has total participation

     Employees has partial participation

**Relationship:** Car Linked to CarReport

**Cardinality:** one to one

**Participation:**

Car has total participation

CarReport has total participation

**Relationship:** Employees searches CarReport

**Cardinality:** many to one

**Participation:**

Employees has total participation

CarReport has total participation

**Relationship:** Customers searches CarReport

**Cardinality:** many to one

**Participation:**

Customers has total participation

CarReport has total participation

**LOGICAL DESIGN WITH NORMAL FORM IDENTIFICATION**

Include your complete updated logical design here. Use the format shown below.

 Table: **EmployeeAccount**

Columns:

**-employeeID [Primary Key]**

*-managerID [Foreign Key - REFERENCES ManagerAccount(managerID)]*

-firstName

-lastName

-email

-phoneNum

-position

*BCNF - Each column has one value, no duplicate columns, key columns are unique, key columns not dependent on non-key columns, all non-key columns are fully dependent on the primary key (no partial dependencies), no dependencies amongst non-key columns, and no transitive functional dependencies.*

Table: **CustomerAccount**

Columns:

**-customerID [Primary Key]**

-firstName

-lastName

-email

-phoneNum

-Address

-Balance

*BCNF - Each column has one value, no duplicate columns, key columns are unique, key columns not dependent on non-key columns, all non-key columns are fully dependent on the primary key (no partial dependencies), no dependencies amongst non-key columns, and no transitive functional dependencies.*

Table: **EmployeeAccountCustomerAccount**

Columns:

*-customerID [Foreign Key - REFERENCES CustomerAccount(customerID)]*

*-employeeID [Foreign Key - REFERENCES EmployeeAccount(employeeID)]*

-messages

-sender

**[Primary Key is composite key (customerID, employeeID)]**

*BCNF - Each column has one value, no duplicate columns, key columns are unique, key columns not dependent on non-key columns, all non-key columns are fully dependent on the primary key (no partial dependencies), no dependencies amongst non-key columns, and no transitive functional dependencies.*

     Table: **CreditCard**

Columns:

**-cardNum [Primary Key]**

-cardHolderName

-expDate

-CVV

-customerID [Foreign Key] References CustomerAccount

*BCNF - Each column has one value, no duplicate columns, key columns are unique, key columns not dependent on non-key columns, all non-key columns are fully dependent on the primary key (no partial dependencies), no dependencies amongst non-key columns, and no transitive functional dependencies.*

     Table: **Cars**

Columns:

**-VINnum [Primary Key]**

-manufacturer

-model

-year

*BCNF - Each column has one value, no duplicate columns, key columns are unique, key columns not dependent on non-key columns, all non-key columns are fully dependent on the primary key (no partial dependencies), no dependencies amongst non-key columns, and no transitive functional dependencies.*

     Table: **Supplier**

Columns:

-**supplierID [Primary Key]**

-name

-phoneNum

-email

*BCNF - Each column has one value, no duplicate columns, key columns are unique, key columns not dependent on non-key columns, all non-key columns are fully dependent on the primary key (no partial dependencies), no dependencies amongst non-key columns, and no transitive functional dependencies.*

Table: **ManagerAccount**

Columns:

-**managerID [Primary Key]**

-firstName

-lastName

-phoneNum

-email

*BCNF - Each column has one value, no duplicate columns, key columns are unique, key columns not dependent on non-key columns, all non-key columns are fully dependent on the primary key (no partial dependencies), no dependencies amongst non-key columns, and no transitive functional dependencies.*

     Table: **SearchCars**

Columns:

-VINnum

-customerID

-employeeID

-**Primary Key (VINnum,customerID,employeeID)**

*BCNF - Each column has one value, no duplicate columns, key columns are unique, key columns not dependent on non-key columns, all non-key columns are fully dependent on the primary key (no partial dependencies), no dependencies amongst non-key columns, and no transitive functional dependencies.*

**E-R Diagram**

Diagram

Description automatically generated

**SQL QUERIES**

List at least **three** SQL queries that perform data retrievals relevant to the features chosen in the current sprint. For each query, paste a **screenshot** of the output, as shown through your user interface.

**SELECT \***

**FROM EmployeeAccountCustomerAccount**

**WHERE customerID = 009 OR customerID = 008 AND employeeID = 004;**

**Graphical user interface, text, application

Description automatically generated**

**SELECT firstName, lastName**

**FROM CustomerAccount AS CA**

**INNER JOIN CreditCard AS CD**

**WHERE CA.customerID = CD.customerID;**

**Graphical user interface, text, application

Description automatically generated**

**SELECT E.firstName, E.lastName**

**FROM EmployeeAccount AS E**

**INNER JOIN ManagerAccount AS M**

**WHERE E.managerID = M.managerID AND E.Position = 'sales';**

**Graphical user interface, text, application

Description automatically generated**

**Sprint 3**

**REQUIREMENTS**

List your updated user stories in decreasing order of priority. Highlight the stories that were completed in Sprint 1 in one color. Highlight the stories that were completed in Sprint 2 in a different color. Highlight the updated/new stories chosen for Sprint 3, if any, in a third color. *There is no need to explicitly show your story refinement process.* Use the format shown below.

|  |  |
| --- | --- |
| **Story ID** | **Story description** |
| Employee | As an employee, I want to search and sell cars to customers |
| Customer | As a customer, I want to search for car details and buy cars from Employees |
| Admin | As an Admin, I want to create and manage accounts for customers and employees |
| Manager | As a manager, I will supervise and oversee my employees and hiring |
| Supplier | As a supplier, I want to supply the dealership with cars and provide number of stock |
| Inventory | Inventory will hold the values for the vehicle details including make, model, VIN |
| Credit Card | Stores credit card information for customers to utilize during transactions |

**CONCEPTUAL DESIGN**

Include your complete updated conceptual design here. Use the format shown below.

 Entity: **EmployeeAccount**

Attributes:

**-employeeID**

**-managerID**

**-firstName**

**-lastName**

**-email**

**-phoneNum**

**-position**

Entity**:** **CustomerAccount**

Attributes:

**-customerID**

**-firstName**

**-lastName**

**-email**

**-phoneNum**

**-Address**

**-Balance**

Entity: **EmployeeAccountCustomerAccount**

Attributes:

**-customerID**

**-employeeID**

**-messages**

**-sender**

Entity: **Credit Card**

Attributes:

**-cardNum**

**-cardHolderName**

**-expDate**

**-CVV**

**-customerID**

Entity: **Cars**

Attributes:

**-VINnum**

**-manufacturer**

**-model**

**-year**

Entity: **Supplier**

Attributes:

**-supplierID**

**-name**

**-phoneNum**

**-email**

Entity: **ManagerAccount**

Attributes:

**-managerID**

**-firstName**

**-lastName**

**-phoneNum**

**-email**

Entity: **SearchCars**

Attributes:

**-VINnum**

**-customerID**

**-employeeID**

**Relationship:** Customer Buy Cars

**Cardinality:** One to Many

**Participation:**

     Customers has total participation

     Cars has partial participation

**Relationship:** Customers Use Credit Card

**Cardinality:** One to Many

**Participation:**

     Customers has total participation

     Credit Card has total participation

**Relationship:** Employees messages Customers

**Cardinality:** Many to Many

**Participation:**

     Employees has total participation

     Customers has total participation

**Relationship:** Manager oversees Employees

**Cardinality:** one to Many

**Participation:**

     Manager has total participation

     Employees has partial participation

**Relationship:** Car Linked to CarReport

**Cardinality:** one to one

**Participation:**

Car has total participation

CarReport has total participation

**Relationship:** Employees searches CarReport

**Cardinality:** many to one

**Participation:**

Employees has total participation

CarReport has total participation

**Relationship:** Customers searches CarReport

**Cardinality:** many to one

**Participation:**

Customers has total participation

CarReport has total participation

**LOGICAL DESIGN WITH HIGHEST NORMAL FORMS AND INDEXES**

Include your complete updated logical design here. Use the format shown below.

Table: **EmployeeAccount**

Columns:

**-employeeID [Primary Key]**

*-managerID [Foreign Key - REFERENCES ManagerAccount(managerID)]*

-firstName

-lastName

-email

-phoneNum

-position

*BCNF - Each column has one value, no duplicate columns, key columns are unique, key columns not dependent on non-key columns, all non-key columns are fully dependent on the primary key (no partial dependencies), no dependencies amongst non-key columns, and no transitive functional dependencies.*

Indexes:

Index 1: clustered

Columns: employeeID

Justification: employeeID is the primary key of the EmployeeAccount table and therefore is a primary/clustered index

Table: **CustomerAccount**

Columns:

**-customerID [Primary Key]**

-firstName

-lastName

-email

-phoneNum

-Address

-Balance

*BCNF - Each column has one value, no duplicate columns, key columns are unique, key columns not dependent on non-key columns, all non-key columns are fully dependent on the primary key (no partial dependencies), no dependencies amongst non-key columns, and no transitive functional dependencies.*

Indexes:

Index 1: clustered

Columns: customerID

Justification: customerID is the primary key of the CustomerAccount table and therefore is a primary/clustered index

Table: **EmployeeAccountCustomerAccount**

Columns:

*-customerID [Foreign Key - REFERENCES CustomerAccount(customerID)]*

*-employeeID [Foreign Key - REFERENCES EmployeeAccount(employeeID)]*

-messages

-sender

**[Primary Key is composite key (customerID, employeeID)]**

*BCNF - Each column has one value, no duplicate columns, key columns are unique, key columns not dependent on non-key columns, all non-key columns are fully dependent on the primary key (no partial dependencies), no dependencies amongst non-key columns, and no transitive functional dependencies.*

Indexes:

Index 1: clustered

Columns: customerID, employeeID

Justification: customerID and employeeID is the composite primary key of the EmployeeAccountCustomerAccount table and therefore is a primary/clustered index

     Table: **CreditCard**

Columns:

**-cardNum [Primary Key]**

-cardHolderName

-expDate

-CVV

-customerID *[Foreign Key - REFERENCES CustomerAccount(customerID)]*

*BCNF - Each column has one value, no duplicate columns, key columns are unique, key columns not dependent on non-key columns, all non-key columns are fully dependent on the primary key (no partial dependencies), no dependencies amongst non-key columns, and no transitive functional dependencies.*

Indexes:

Index 1: clustered

Columns: cardNum

Justification: cardNum is the primary key of the CreditCard table and therefore is a primary/clustered index

Index 2: nonclustered

Columns: customerID

Justification: customerID is a foreign key that references the CustomerAccount table. So, employees might want to find out which credit card belongs to which customer when completing purchases. Therefore it is a nonclustered index

     Table: **Cars**

Columns:

**-VINnum [Primary Key]**

-manufacturer

-model

-year

*BCNF - Each column has one value, no duplicate columns, key columns are unique, key columns not dependent on non-key columns, all non-key columns are fully dependent on the primary key (no partial dependencies), no dependencies amongst non-key columns, and no transitive functional dependencies.*

Indexes:

Index 1: clustered

Columns: VINnum

Justification: VINnum is the primary key of the Cars table and therefore is a primary/clustered index

Index 2: nonclustered

Columns: manufacturer, model

Justification: manufacturer and model are all in order of specificity of a car's identity. So, when searching for a specific car a user might want to narrow the results down to the manufacturer and model. This is why having an index on these two columns would speed up query times greatly.

     Table: **Supplier**

Columns:

-**supplierID [Primary Key]**

-name

-phoneNum

-email

*BCNF - Each column has one value, no duplicate columns, key columns are unique, key columns not dependent on non-key columns, all non-key columns are fully dependent on the primary key (no partial dependencies), no dependencies amongst non-key columns, and no transitive functional dependencies.*

Indexes:

Index 1: clustered

Columns: supplierID

Justification: supplierID is the primary key of the Supplier table and therefore is a primary/clustered index

Index 2: nonclustered

Columns: name

Justification: employers might frequently want to search for suppliers by name. Indexing it would speed up query times

     Table: **ManagerAccount**

Columns:

-**managerID [Primary Key]**

-firstName

-lastName

-phoneNum

-email

*BCNF - Each column has one value, no duplicate columns, key columns are unique, key columns not dependent on non-key columns, all non-key columns are fully dependent on the primary key (no partial dependencies), no dependencies amongst non-key columns, and no transitive functional dependencies.*

Indexes:

Index 1: clustered

Columns: managerID

Justification: managerID is the primary key of the ManagerAccount table and therefore is a primary/clustered index

     Table: **SearchCars**

Columns:

-VINnum

-customerID  *[Foreign Key - REFERENCES CustomerAccount(customerID)]*

-employeeID *[Foreign Key - REFERENCES EmployeeAccount(employeeID)]*

-**Primary Key (VINnum,customerID,employeeID)**

*BCNF - Each column has one value, no duplicate columns, key columns are unique, key columns not dependent on non-key columns, all non-key columns are fully dependent on the primary key (no partial dependencies), no dependencies amongst non-key columns, and no transitive functional dependencies.*

Indexes:

Index 1: clustered

Columns: VINnum, customerID, employeeID

Justification: VINnum, customerID, and employeeID is the composite primary key of the SearchCars table and therefore is a primary/clustered index

**VIEWS AND STORED PROGRAMS**

List the views relevant to your application here. Use the format specified below.

**View**: view\_customer\_credit\_card

Goal: The goal of this view is to show the customer’s ID along with their card numbers

**View**: view\_employeeID\_info

Goal: The goal of this view is to show the first name, last name, phone number, and email of all employees

**View**: view\_customer\_balance

Goal: The goal of this view is to show all information for customers who have a balance that needs to be paid.

List the stored programs relevant to your application thus far here. Use the format specified below for the different kinds of stored programs. Note: if you do not have a particular type of stored program in your application, just leave that part out.

**Stored procedure**: show\_car\_inventory

Parameters: (IN manu varchar(20), IN mo varchar(20))

Goal: The goal of this procedure is to search for cars by their manufacturer and model

**Stored procedure**: show\_messages

Parameters: (IN cust int, IN empl int)

Goal: The goal of this procedure is to search for specific messages between employees and customers

**Stored procedure**: show\_customer\_information

Parameters: (IN cust int)

Goal: The goal of this procedure is to search for customer’s information in the database

**Views Screenshots**

1. view\_customer\_credit\_card

Text

Description automatically generated with low confidence

Graphical user interface, text, application, Word

Description automatically generated

1. view\_employeeID\_info

Graphical user interface, text, application

Description automatically generated

Text

Description automatically generated

1. view\_customer\_balance

Calendar

Description automatically generated with medium confidence

Graphical user interface, text, application, table

Description automatically generated

**Procedures Screenshots**

1. show\_car\_inventory

**Graphical user interface, application

Description automatically generated**

**Graphical user interface, text, application

Description automatically generated**

1. show\_customer\_information

**Graphical user interface

Description automatically generated**

**Graphical user interface, text, application

Description automatically generated**

1. show\_messages

**Graphical user interface

Description automatically generated**

**Graphical user interface, text, application

Description automatically generated**