Introduction to Smooth Driving College Of Motoring

**Abu Bakar Siddik(19897)**

**Tesfaldet Yemaneab(19880)**

**Simon Gezae(19830)**

**Zainab Salia(19878)**

**San Francisco Bay University**

**Course Number: CS457L**

Scenario Description

Established in Glasgow in 1992, the Smooth Drive College of Motoring has grown to include multiple offices in strategic cities around Scotland. The management of the school is being hampered by the growing number of students and the inadequate communication across offices as the school expands. The business began recruiting extra administrators to address this. Nonetheless, another issue that has to be addressed is the inadequate communication even within the same city. Director Dave MacLeod understands that in order to improve information sharing and streamline operations, a comprehensive database system is urgently needed. Dave has enlisted our team's experience to create a database system that is specifically suited to the needs of the school in order to address these problems.

The school plan to create database system to store and manage data of the administrative staff. In addition to that, the database system needs to store the client information. Moreover, Individual lesson, instructor record, car detail, driving test result will also be stored in the database.

Abstract

The case study is about a smooth drive collage of motoring in Glasgow Scotland. In this case study as the company gets bigger and bigger, the communication and sharing of information between offices even in the same city became difficult. So, the school director Dave Macleod feels that his company will be short-lived if this persists, and he wants a remedy for that. Then he approached us to create a database for his company. The database will be able to store information about the instructors, the administrative staff, the clients, and the test schedule, register for recording all the necessary info about the clients' application forms, and provide a way to assess the client by having an interview session, provide a way the client can request a particular instructor and also apple changes request any time during the process, he also needs us to provide a way a client can book a single lesson or a block of lessons with a cost discount. The can be arranged as early as 8 am and no later than 7 pm as the duration of the class session is 1 hour and the office closes by 8 pm.

To solve this issue, we will start with data modeling by identifying the entities, their attributes, and the relationships between them. In this case, the main entities are instructors, office, clients, managers, cars, mileage and classSessions, test schedules. Attributes for each entity need to be defined, such as OfficeNumber, location, city, zipcode, phoneNumber for an office entity; AdministratorId, name, address, PhoneNumber, ManagerId, officeNumber for an administrator, SessionId schedule, for a car entity; roomNumber, IstructorId for a class session, PlateNumber, yearBuilt, type, mileageCoverd for each car as an entity; Name, ClientId, Address, and Contact details for each client; name, classSchedule, instructorId , address, officeNumber for the instructors; mileages can have maintenance, engineSize, weightOfCar, fuelEfficiency as an atrributes.

The next step is database design, where the structure and organization of the database are determined. This includes defining tables for each entity and establishing relationships between them. Primary keys, such as PlateNumber for cars and ClientID for clients, will be used to uniquely identify each record in the database. Foreign keys will be utilized to establish relationships between tables, such as linking instructors to classSessions, carsAssigned to them and clients.

After the design phase, the database implementation process begins. This involves creating the database schema and tables, defining data types and constraints, and implementing the necessary queries and transactions to interact with the database. The database management system will depend on the specific requirements and technologies chosen for this project.

Overall, this project aims to develop a new and improved database management system for Smooth Drive College of Motoring to address the current system's inefficiencies. By following the key steps of data modeling, database design, implementation, and testing, the new system will provide a more efficient and effective means of storing and managing information about cars, clients, instructors, interview schedules, training Session, and test schedules. This will enable Smooth Drive Collage of Motoring to enhance its operations, streamline processes, and ultimately provide better service to its clients.

**Step 1.1. Write out the format of every source document that provides information to be stored in the database.**

**Client Reg. Form:**

Client ID: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

First Name: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Last Name: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Address: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_City: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Contact No: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Postal Code: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Age: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Gender: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Date of Birth: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_E-mail: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Provisional Driving License: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Car Preference: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Preferred Instructor: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Lesson Type: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Lesson Start time: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Lesson End Time: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Interview Record (Instructor-Client):**

Instructors First Name: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Instructors Last Name: \_\_\_\_\_\_\_\_\_\_\_\_\_

Client’s First Name: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Client’ Last Name: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Client’s Required Service: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Interview Date: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Interview Time: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Assessment: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Provisional License Validation: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Lesson Record Form:**

Client First Name: ­­­­­­­­­­­­­­­­­\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Client’s Last Name: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Instructor First Name: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Instructor Last Name: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Car Used: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Mileage: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Lesson Start Time: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Lesson End Time: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Lesson Date: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Lesson Progress: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Car Inspection Report:**

Car ID: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Inspector First Name: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Inspector Last Name: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Inspection Date: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Fault Description: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Driving Test Application Form:**

Client’s ID: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Client First Name: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Client Last Name: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Test Date: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Test Time: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Test Type: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Test Preparation Details: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Client Test Results:**

Client First Name: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Client Last Name: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Test Date: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Test Time: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Test Result: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Reason for Failure: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Step 1.2 Write out the format of every routine report to be produced using the database. Do not make any assumptions about how records actually appear in the database, or what programming may be involved in producing the report**.

**Office Report:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Office Add. | Office Manager | Total Staff | Total Clients | Contact Info(Phone) | Office Hours |
| 123 olive a, | Siddik | 15 | 100 | 123456789 | 9-5:00 pm |
| XXXX | XXX | XXXX | XXXX | XXXXX | XXXXXX |
|  |  |  |  |  |  |

**Instructor Report:**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Instructor Name | Gender | License Type | Assigned Car | Total Clients | Contact Info(Phone) | Availability |
| Siddik | Male | XXXX | XXXXX | 30 | 123456789 | XXXXXX |
| XXXX | XXXXX | XXXXX | XXXXX | XXXXX | XXXXX | XXXXX |
|  |  |  |  |  |  |  |

**Lesson Schedule Report:**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Lesson Date | Lesson Time | Instructor Name | Client Name | Lesson Type | Duration | Location | Lesson Mileage | Test Date | Test Locatio |
| 1/1/23 | 10 am | Siddik | Simon | XXXX | XXXX | XXXXX | XXXXx | XX | XXX |
| XXX | XXXX | XXX | XXX | XXX | XXX | XXXX | XXX | XXX | XX |
|  |  |  |  |  |  |  |  |  |  |

**Client Registration Report:**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Client Name | Gender | D.O.B | Address | Contact  Info | License  Type | Instructor  Preference | Assigned  Instructor | Total  Lesson | Test  His. |
| Tes | Male | XXX | XXXX | 12345 | XX | Car | Siddik | 10 | Pass |
| XXX | XXX | XXX | XXXX | XXXXX | XX | XXX | XXXXXX | XX | XXX |
|  |  |  |  |  |  |  |  |  |  |

**Car Maintenance Report:**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Car Number | Assigned Instructor | Inspection Date | Inspection Type | Inspection Details | Maintenance  Schedule | Car  Availability |
| BOX123 | Siddik | XX XX XXX | XX XXXX | XXXXX | XXXXX | XXX |
| XXX | XXXX | XXXX | XXXX | XXXX | XXXX | XXXX |
|  |  |  |  |  |  |  |

**Step 1.4 Write out an initial list of assumptions for the project.**

* Instructors are assigned to cars and clients according to the clients' choices and availability.
* Clients should do proper practice with an instructor. And an instructor will guide the client accordingly.
* Clients are responsible for applying for driving tests when they are ready.
* Test results are recorded by an external testing center.
* Instructors are responsible for preparing clients for both the written and practical parts of the driving test.
* Each office has staff members with specific roles, such as Managers, Instructors, and administrative staff.
* Clients need to make sure that the information they are providing is accurate.
* As clients are sharing their personal information, the company will make sure that their system is safe enough.

After the test, the test report should be recorded no matter whether they pass or fail.

**Step 2.1 Designing the Data Dictionary**

Address - The administrator's actual location, including street address, city, and zip code.

Address - The client's actual location, including street address, city, and zipcode.

AdministratorId - Each administrator is provided a unique identification, which aids in the differentiation and retrieval of specific administrative records inside the database.

AdministratorName - The name of the administrator in charge of administrative responsibilities, which helps with personal identification and communication.

Age: age of the instructor.

ApplicationId: A one-of-a-kind identification for each application form that aids in tracking and retrieval.

AssessmentResults: The outcome of the client's driving abilities interview evaluation.

AuditTrail: information on the person who made the update and when.

Availability: What are the available times for having the session.

calculationMethod: the method we use to calculate the difference between current and previous records

CarReg.Num: Their car registration number.

City: The city where the instructor is working now

ClassSchedule: It means when the class time is booked.

classSessionDuration: the duration of the class session

classSessoininformationForm: a form that contains information on the class session

ClientAge: the age of the client

ClientId - Each client is provided a unique identification, which allows the database to differentiate and maintain individual student records.

ClientID: (Foreign Key) Identifier linking the test record to a specific client.

ClientId: A foreign key that associates the interview session with a certain client.

ClientId: A foreign key that connects the application form to the individual client.

ClientName - The student's name, which serves as a personal identity and facilitates contact.

ClientName: name of the clients enrolled in the class session

Comments: Additional remarks or notes from the interview session

Contact details - Phone numbers and email addresses that enable connection with the customer, allowing good communications between the driving school and its pupils.

ContactNumber: Contact number of the manager.

County: It is the county where the instructor works in now.

D.O.B: Birthdate of the instructor.

Date: The date and time of the driving session.

Description: the total distance traveled by a specific car

Duration: The driving session's duration in hours.

Email: Email address of the manager.

EngineSize: the size of the engine fo a specific car

ExperienceLevel: how much experience they have.

FailureReason: If applicable, reasons for the client's failure.

FuelEfficiency: how much fuel a car consumes per mile.

FullTime: Working more than or equal to 40 hours a week.

Gender: gender of the instructor.

HireDate: when the instructor is hired.

InstructorAge: age of the instructor for the class session

InstructorAssigned: The teacher who was allocated to the customer based on the interview.

InstructorID: (Foreign Key) Identifier linking the test record to a specific instructor.

InstructorId: A foreign key linking the driving session to the instructor in charge.

InstructorID: Every instructor has a unique id to identify them.

instructorId: id of the instructor assigned for a specific session

InstructorName: name of the instructors assigned for the class session

InterviewId: A one-of-a-kind identification for each interview session, used for tracking and retrieval.

JobType: It means is the instructor working part time or full time here.

lastUpdated: information on the date and time when the last record of mileage was updated.

LicenseType: Which type of license does the instructor have.

Maintenance: the number of maintenances a car has so far

MaintenanceDate: The date when maintenance tasks were completed.

MaintenanceDetails: Specifics about the maintenance operations performed.

MaintenanceId: Each maintenance record has a unique identification, which aids in tracking and retrieval.

MaintenanceType: Describes the sort of maintenance conducted (for example, routine maintenance, repairs, and inspections).

ManagerId - A unique identity that connects an administrator to a manager, providing a hierarchical relationship inside the driving school's administrative structure.

ManagerID: (Primary Key) Unique identifier for each manager.

MileageCovered - The overall distance driven by the automobile, recording data on vehicle usage and wear during driving lessons.

mileageInformationForm: is a form that contains mileage information.

Name: Full name of the manager.

Name: It means the instructor name.

NoOfSessions: the number of sessions a client took

OfficeCity: which city is the office location

OfficeID: (Foreign Key) Identifier indicating the office where the manager is responsible.

OfficeID: unique id of the office

OfficeLocation: the location of the office it contains street address zipcode

OfficeName: name of the office

OfficeNumber - Each office inside the driving school has a unique identity that indicates the precise location or branch to which an administrator is allocated.

OfficeNumber: Every instructor has an office number as a contact.

OfficePhoneNumber: phone number of the office

PartTime: Working less than 40 hours in a week

PhoneNumber - The administrator's contact number, which allows for communication and cooperation with the personnel.

PlateNnumber: a unique identification number assigned for a specific car

PlateNumber - Each automobile is issued a unique identity, which is used for vehicle identification and tracking in the database.

PlateNumber: A foreign key that connects the driving experience to the individual vehicle utilized.

PlateNumber: A foreign key that connects the maintenance record to the individual automobile to which it refers.

PreferredInstructor: The client's preferred teacher for driving instruction, if any.

PreferredSchedule: Desired driving lesson schedule or time.

PreferredVehicleForPractise: the car type the client want for a practical session and for a test also

purposeOfRecord: why we are taking the records of our mileage.

Responsibilities: Description of the day-to-day responsibilities and duties of the manager.

Role: Designation of the manager (Manager, Senior Instructor).

RoomNumber: is the roomnumber assigned for a specific session

schedule: assigning a specific time and days to go on with the sessions

SessionId: A distinct identity for each driving session, allowing for easier monitoring and retrieval.

sessionId: it means a session should have a unique number to identify it

SpecialRequests: Any extra requests or preferences supplied by the customer are referred to as SpecialRequests.

StaffID: (Foreign Key) Identifier linking the manager to the staff table.

staffIdOfREcorder: information of the staff who took the record of the mileage.

statusOfMileage: information on what status the mileage is e.g pending

TestDate: Date of the driving test.

TestID: (Primary Key) Unique identifier for each test record.

TestResult: Boolean indicating whether the client passed the test.

theoryClass: the class that a student takes before a knowledge test is conducted

Type - The car's classification or category, indicating the kind of vehicle used for driving lessons (e.g., sedan, hatchback, SUV).

VehicleType: what are the types of vehicle that the instructor can drive.

weightOfCar: the weight of a car

YearBuilt - The year the automobile was produced, providing information on the vehicle's age.

ZipCode: Area zip code of the location

**Step 2.2 Modify the List of Assumptions as needed**

* Each office is uniquely identified by its OfficeNumber.
* Each car is uniquely identified by its PlateNumber.
* Each client is uniquely identified by their ClientId.
* Each instructor is uniquely identified by their InstructorId.
* Each class session is uniquely identified by its SessionId.
* Each administrator is uniquely identified by their AdministratorId.
* Each test schedule has unique attributes that need to be defined.

**Step 2.3 Write out a cross-reference table, showing what data items appear on what forms, reports, or transactions.**



**Step 3.1: Make a list of all entities and their associated attributes.**

**Instructors** (InstructorId, Name, ClassSchedule, Address, OfficeNumber)

OfficeNumber is a foreign key which is referencing Office.

**Office** (OfficeNumber, Location, City, Zipcode, PhoneNumber)

**Administrators** (AdministratorId, Name, Address, PhoneNumber, ManagerId, OfficeNumber)

ManagerId is a Foreign Key referencing Managers && OfficeNumber is a Foreign Key referencing Office.

**Managers** (ManagerId, Name, Address, PhoneNumber)

**ClassSessions** (SessionId, Schedule, InstructorId, RoomNumber)

InstructorId is a Foreign Key referencing Instructors.

**Cars** (PlateNumber, YearBuilt, Type, MileageCovered, Maintenance, EngineSize, WeightOfCar, FuelEfficiency, InstructorId)

InstructorId is a Foreign Key that refer Instructors.

**Clients** (ClientId, Name, Address, ContactDetails)  
**TestSchedules** (TestScheduleId, Schedule, ClientId)

ClientId is a Foreign Key referencing Clients.

**Mileage** (MileageId, Maintenance, EngineSize, WeightOfCar, FuelEfficiency, PlateNumber)

PlateNumber is a Foreign key referencing Cars.

**ApplicationForm** (FormId, Details, ClientId)

ClientId is a Foreign key which refer Client.

**Step 3.2: Make a list of relationships to be represented, and any descriptive attributes for them.**

Instructor Teaches ClassSession: Many-to-Many

Car AssignedTo Instructor: Many-to-Many

ClassSession ConductedBy Instructor: Many-to-One

Administrator Manages Office: One-to-One

Client RequestsInstructor Instructor: Many-to-Many

Client Books ClassSession: Many-to-Many

Client ScheduledFor TestSchedule: One-to-Many

Mileage BelongsTo Car: One-to-One

Extended Diagram with Generalization and Specialization:

**Generalization/Specialization:**

Person:

Attributes: PersonId (Primary Key), name, address, phoneNumber.

*Instructor (Specialization of Person):*

Attributes: InstructorId, classSchedule, officeNumber.

*Client (Specialization of Person):*

Attributes: ClientId, bookedLessons.

*Administrators (Specialization of Person):*

Attributes: (AdministratorId, ManagerId, OfficeNumber)

Disjoint:

Disjoint the Person entity into separate entities for Administrator, Instructor, and Client to ensure each person is exclusively one of these types.

**Step 3.3: Draw an E-R diagram to represent the enterprise. Be sure to identify  
relationship participation and cardinality constraints, any weak entity sets, and rolenames, if needed.**

A diagram of a company

Description automatically generated

**LucidChart Link:** [https://lucid.app/lucidchart/51851103-f7d1-40f2-b68b-1813341add6d/edit?invitationId=inv\_4b31ba1d-0dca-42f1-bec8-54e3ff485a79&page=0\_0#](https://lucid.app/lucidchart/51851103-f7d1-40f2-b68b-1813341add6d/edit?invitationId=inv_4b31ba1d-0dca-42f1-bec8-54e3ff485a79&page=0_0)

**Step 3.4 Update on the list of assumptions as:**

* Instructors are assigned to cars and clients based on the clients' preferences and instructor availability.
* Clients have the flexibility to choose their preferred instructor, and assignments are made based on availability.
* Clients should engage in proper practice sessions guided by their assigned instructors.
* Clients are responsible for applying for driving tests when they feel ready.
* Test results, whether pass or fail, are recorded by an external testing center.
* Instructors are responsible for preparing clients for both the written and practical parts of the driving test.
* Each instructor is uniquely identified by their InstructorId
* Each office has staff members with specific roles, including Managers, Instructors, and administrative staff.
* Each office is uniquely identified by its OfficeNumber.
* Clients are responsible for providing accurate information.
* The company ensures the security of the system to protect the personal information shared by clients.
* After the test, whether a client passes or fails, the test report is recorded.
* Each car is uniquely identified by its PlateNumber.
* Each client is uniquely identified by their ClientId.
* Each class session is uniquely identified by its SessionId.
* Each administrator is uniquely identified by their AdministratorId.
* Each test schedule has unique attributes that need to be defined.

**Step 3.5 Write the schema for the database, but do not create the actual database at this stage.**

**Instructors** (InstructorId, Name, ClassSchedule, Address, *OfficeNumber*)

OfficeNumber is a foreign key which is referencing Office.

**Office** (OfficeNumber, Location, City, Zipcode, PhoneNumber)

**Administrators** (AdministratorId, Name, Address, PhoneNumber, *OfficeNumbe, ManagerId*)

ManagerId is a Foreign Key referencing Managers & OfficeNumber is a Foreign Key referencing Office.

**Managers** (ManagerId, Name, Address, PhoneNumber)

**ClassSessions** (SessionId, Schedule, *InstructorId*, RoomNumber)

InstructorId is a Foreign Key referencing Instructors.

**Cars** (PlateNumber, YearBuilt, Type, MileageCovered, *InstructorId*)

InstructorId is a Foreign Key that refer Instructors.

**Clients** (ClientId, Name, Address, ContactDetails, *InstructorId*)

InstructorId is a Foreign Key referencing Instructors  
**TestSchedules** (TestScheduleId, Schedule, *ClientId*)

ClientId is a Foreign Key referencing Clients.

**Mileage** (MileageId, Maintenance, EngineSize, WeightOfCar, FuelEfficiency, *PlateNumber*)

PlateNumber is a Foreign key referencing Cars.

**ApplicationForm** (FormId, Details, *ClientId*)

ClientId is a Foreign key which refer Client.

**Step 4.1. Map the ER model developed at the end of Module 3 to a relational model.**

Instructor

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| InstructorID | Name | ClassSchedule | Address | OfficeNumber |
| 1 | John Smith | Monday 9am-12pm, Wednesday 2pm-5pm | 123 Main St, City, Country | A101 |
| 2 | Emily Johnson | Tuesday 10am-1pm, Thursday 3pm-6pm | 456 Elm St, City, Country | B205 |
| 3 | Michael Davis | Monday 1pm-4pm, Friday 9am-12pm | 789 Oak St, City, Country | C303 |
| 4 | Sarah Wilson | Wednesday 10am-1pm, Friday 2pm-5pm | 101 Pine St, City, Country | D115 |
| 5 | James Miller | Tuesday 9am-12pm, Thursday 2pm-5pm | 222 Cedar St, City, Country | E408 |
| 6 | Olivia Brown | Monday 10am-1pm, Wednesday 3pm-6pm | 333 Maple St, City, Country | F511 |
| 7 | William Garcia | Wednesday 1pm-4pm, Friday 10am-1pm | 444 Walnut St, City, Country | G607 |
| 8 | Ava Martinez | Tuesday 1pm-4pm, Thursday 9am-12pm | 555 Oak St, City, Country | H709 |
| 9 | Liam Jones | Monday 9am-12pm, Friday 2pm-5pm | 666 Pine St, City, Country | I813 |

Office

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| OfiiceNumber | Location | City | Zipcode | PhoneNumber |
|  |  |  |  |  |

Administrators

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| AdministratorID | Name | Address | PhoneNumber | OfficeNumber | ManagerID |
|  |  |  |  |  |  |

Manager

|  |  |  |  |
| --- | --- | --- | --- |
| ManagerID | Name | Address | PhoneNumber |
|  |  |  |  |

ClassSession

|  |  |  |  |
| --- | --- | --- | --- |
| SessionID | Schedule | InstructorID | RoomNumber |
|  |  |  |  |

Cars

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| PlateNumber | yearBuilt | Type | MileageCovered | InstructorID |
|  |  |  |  |  |

Client

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ClientID | Name | Address | Address | InstructorID |
|  |  |  |  |  |

TestSchedules

|  |  |  |
| --- | --- | --- |
| TestScheduleID | Date | ClientID |
|  |  |  |

ApplicantForm

|  |  |  |
| --- | --- | --- |
| FormID | Details | ClientID |
|  |  |  |

**Step 5.1 Update the data dictionary and list of assumptions as needed. For each table, write the table name and write out the names, data types, and sizes of all the data items, and identify any constraints, using the conventions of the DBMS you will use for implementation.**

CREATE TABLE Instructors (

InstructorID INT PRIMARY KEY,

Name VARCHAR(255),

ClassSchedule VARCHAR(255),

Address VARCHAR(255),

OfficeNumber INT,

FOREIGN KEY (OfficeNumber) REFERENCES Office(OfficeNumber)

);

CREATE TABLE Office (

OfficeNumber INT PRIMARY KEY,

Location VARCHAR(255),

City VARCHAR(255),

Zipcode INT(5),

PhoneNumber VARCHAR(20)

);

CREATE TABLE Administrators (

AdministratorID INT PRIMARY KEY,

Name VARCHAR(255),

Address VARCHAR(255),

PhoneNumber VARCHAR(20),

OfficeNumber INT,

FOREIGN KEY (OfficeNumber) REFERENCES Office(OfficeNumber),

FOREIGN KEY (ManagerID) REFERENCES Managers(ManagerID)

);

CREATE TABLE Managers (

ManagerID INT PRIMARY KEY,

Name VARCHAR(255),

Address VARCHAR(255),

PhoneNumber VARCHAR(20)

);

CREATE TABLE ClassSessions (

SessionID INT PRIMARY KEY,

Schedule VARCHAR(255),

InstructorID INT,

FOREIGN KEY (InstructorID) REFERENCES Instructors(InstructorID),

RoomNumber VARCHAR(255)

);

CREATE TABLE Cars (

PlateNumber VARCHAR(255) PRIMARY KEY,

YearBuilt INT,

Type VARCHAR(255),

MileageCovered INT,

InstructorID INT,

FOREIGN KEY (InstructorID) REFERENCES Instructors(InstructorID)

);

CREATE TABLE Clients (

Clientid INT PRIMARY KEY,

Name VARCHAR(255),

Address VARCHAR(255),

ContactDetails VARCHAR(255),

InstructorID INT,

FOREIGN KEY (InstructorID) REFERENCES Instructors(InstructorID)

);

CREATE TABLE TestSchedules (

TestScheduleID INT PRIMARY KEY,

Schedule VARCHAR(255),

Clientid INT,

FOREIGN KEY (Clientid) REFERENCES Clients(Clientid)

);

CREATE TABLE Mileage (

Mileageld INT PRIMARY KEY,

Maintenance VARCHAR(255),

EngineSize INT,

WeightOfCar INT,

FuelEfficiency INT,

PlateNumber VARCHAR(255),

FOREIGN KEY (PlateNumber) REFERENCES Cars(PlateNumber)

);

CREATE TABLE ApplicationForm (

ApplicationID INT PRIMARY KEY,

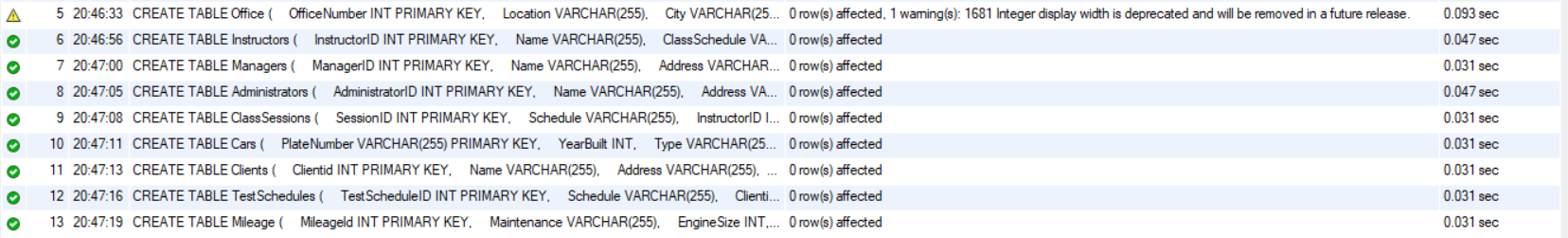
Details VARCHAR(200),

ClientId INT,

FOREIGN KEY (ClientId) REFERENCES Clients(ClientId)

);

**Step 5.2 Write and execute SQL statements to create all tables needed to implement the design.**



**Step 5.3 Insert at least ten records in each table, preserving all constraints. Put in enough data to demonstrate how the database will function.**

INSERT INTO Office (OfficeNumber, Location, City, Zipcode, PhoneNumber)

VALUES

(1, 'Main Office', 'Cityscape', 89923, '787-878-3333'),

(2, 'Branch Office', 'Markshire', 67890, '987-654-3210'),

(3, 'Another Branch Office', 'Mark', 67990, '917-654-3211'),

(4, 'Main Office', 'Cityscape', 45039, '990-989-2343'),

(5, 'Main Office', 'New City', 76767, '982-233-2433'),

(6, 'Branch Office', 'Marskhire', 78278,'565-959-9633'),

(7, 'Tech Park Office', 'Tech City', 76543, '777-777-7777'),

(8, 'Industrial Office', 'Industry Town', 87654, '888-888-8888'),

(9, 'Rural Office', 'Countryside', 34567, '999-999-9999'),

(10, 'Harbor Office', 'Harbor City', 23456, '000-000-0000');

INSERT INTO Managers (ManagerID, Name, Address, PhoneNumber)

VALUES

(11, 'Joseph', '123 Harbour St', '111-222-3333'),

(12, 'Jane', '456 Skeeda St', '444-555-6666'),

(13, 'Jake', '789 Octopuel St', '777-888-9999'),

(14, 'Dalia', '321 Skeeda St', '111-999-7777'),

(15, 'Frd', '654 Harbour St', '333-444-5555'),

(16, 'Jessica', '987 Stragash St', '222-666-4444'),

(17, 'Ohana', '234 Stragash St', '555-333-2222'),

(18, 'Joan', '567 Skeeda St', '888-777-6666'),

(19, 'Tony', '876 Cottontail St', '999-555-4444'),

(20, 'Jill', '345 Taoshi St', '000-111-2222');

INSERT INTO Instructors (InstructorID, Name, ClassSchedule, Address, OfficeNumber)

VALUES

(21, 'Blub', 'Morning Classes', '789 Oogway St', 1),

(22, 'Alicia', 'Afternoon Classes', '321 Uncharted St', 2),

(23, 'Schwarz', 'Evening Classes', '654 Telekin St', 3),

(24, 'Diana', 'Weekend Classes', '987 Jogesh St', 4),

(25, 'Ed', 'Online Classes', '234 Andher St', 5),

(26, 'Shrek', 'Special Workshops', '567 Citytip St', 6),

(27, 'Gingery', 'Intensive Courses', '876 Oogway St', 7),

(28, 'Holland', 'Accelerated Programs', '345 Shifu St', 8),

(29, 'Hemsworth', 'Flexible Schedules', '901 Telekin St', 9),

(30, 'Justin', 'Customized Training', '112 Mala St', 10);

INSERT INTO Administrators (AdministratorID, Name, Address, PhoneNumber, OfficeNumber, ManagerID)

VALUES

(31, 'Admin One', '111 Training St', '111-111-1111', 1, 11),

(32, 'Admin Two', '222 Andher St', '222-222-2222', 2, 12),

(33, 'Admin Three', '333 Jogesh St', '333-333-3333', 3, 13),

(34, 'Admin Four', '444 Tiana St', '444-444-4444', 4, 14),

(35, 'Admin Five', '555 Hampton St', '555-555-5555', 5, 15),

(36, 'Admin Six', '666 Milpitas', '666-666-6666', 6, 16),

(37, 'Admin Seven', '777 Stan St', '777-777-7777', 7, 17),

(38, 'Admin Eight', '888 Ford St', '888-888-8888', 8, 18),

(39, 'Admin Nine', '999 Jiggle St', '999-999-9999', 9, 19),

(40, 'Admin Ten', '000 Ten St', '000-000-0000', 10, 20);

INSERT INTO ClassSessions (SessionID, Schedule, InstructorID, RoomNumber)

VALUES

(41, 'Monday 9 AM', 21, 'Room 101'),

(42, 'Wednesday 2 PM', 22, 'Room 202'),

(43, 'Friday 10 AM', 23, 'Room 303'),

(44, 'Tuesday 4 PM', 24, 'Room 404'),

(45, 'Thursday 1 PM', 25, 'Room 505'),

(46, 'Saturday 11 AM', 26, 'Room 606'),

(47, 'Monday 6 PM', 27, 'Room 707'),

(48, 'Wednesday 3 PM', 28, 'Room 808'),

(49, 'Friday 12 PM', 29, 'Room 909'),

(50, 'Tuesday 8 AM', 30, 'Room 1001');

INSERT INTO Cars (PlateNumber, YearBuilt, Type, MileageCovered, InstructorID)

VALUES

('ABC123', 2019, 'Sedan', 20000,21),

('XYZ789', 2020, 'SUV', 15000,22),

('LMN456', 2018, 'Hatchback', 18000,23),

('PQR789', 2021, 'Convertible', 12000,24),

('JKL321', 2017, 'Coupe', 25000,25),

('DEF456', 2019, 'Sedan', 22000,26),

('GHI789', 2022, 'SUV', 13000,27),

('MNO123', 2016, 'Truck', 30000,28),

('UVW456', 2020, 'Van', 16000,29),

('RST789', 2015, 'Sedan', 28000,30);

INSERT INTO Clients (Clientid, Name, Address, ContactDetails, InstructorID)

VALUES

(51, 'Client One', '111 St', '111-111-1111',21),

(52, 'Client Two', '222 St', '222-222-2222',22),

(53, 'Client Three', '333 St', '333-333-3333',23),

(54, 'Client Four', '444 St', '444-444-4444',24),

(55, 'Client Five', '555 St', '555-555-5555',25),

(56, 'Client Six', '666 St', '666-666-6666',26),

(57, 'Client Seven', '777 St', '777-777-7777',27),

(58, 'Client Eight', '888 St', '888-888-8888',28),

(59, 'Client Nine', '999 St', '999-999-9999',29),

(60, 'Client Ten', '000 St', '000-000-0000',30);

INSERT INTO TestSchedules (TestScheduleID, Schedule, Clientid)

VALUES

(61, 'Friday 3 PM',51),

(62, 'Saturday 10 AM',52),

(63, 'Thursday 2 PM',53),

(64, 'Monday 11 AM',54),

(65, 'Wednesday 5 PM',55),

(66, 'Saturday 1 PM',56),

(67, 'Tuesday 9 AM',57),

(68, 'Friday 4 PM',58),

(69, 'Monday 2 PM',59),

(70, 'Thursday 11 AM',60);

INSERT INTO Mileage (Mileageld, Maintenance, EngineSize, WeightOfCar, FuelEfficiency, PlateNumber)

VALUES

(71, 'Regular Checkup', 2.0, 3000, 25, 'ABC123'),

(72, 'Oil Change', 2.5, 3500, 22, 'XYZ789'),

(73, 'Tire Rotation', 2.2, 3200, 24, 'LMN456'),

(74, 'Brake Inspection', 2.8, 3800, 20, 'PQR789'),

(75, 'Air Filter Replacement', 2.1, 3100, 26, 'JKL321'),

(76, 'Transmission Flush', 2.3, 3300, 23, 'DEF456'),

(77, 'Coolant Replacement', 2.4, 3400, 21, 'GHI789'),

(78, 'Battery Check', 2.7, 3700, 19, 'MNO123'),

(79, 'Alignment', 2.6, 3600, 27, 'UVW456'),

(80, 'Spark Plug Replacement', 2.9, 3900, 18, 'RST789');

INSERT INTO ApplicationForm (ApplicationID, Details, ClientId)

VALUES

(1, 'Application Details 1', 51),

(2, 'Application Details 2', 52),

(3, 'Application Details 3', 53),

(4, 'Application Details 4', 54),

(5, 'Application Details 5', 55),

(6, 'Application Details 6', 56),

(7, 'Application Details 7', 57),

(8, 'Application Details 8', 58),

(9, 'Application Details 9', 59),

(10, 'Application Details 10', 60);

**Step 5.4 Write SQL statements that will process five non-routine requests for information from the database just created. For each, write the request in English, followed by the corresponding SQL command.**

1. Request: List all administrators and their associated managers:

SELECT Administrators.Name AS Administrator, Managers.Name AS Manager

FROM Administrators

JOIN Managers ON Administrators.ManagerID = Managers.ManagerID;

1. Request: Find the class schedule, room number, and instructor name for sessions scheduled on Wednesday:

SELECT ClassSessions.Schedule, ClassSessions.RoomNumber, Instructors.Name AS InstructorName

FROM ClassSessions

JOIN Instructors ON ClassSessions.InstructorID = Instructors.InstructorID

WHERE DAYNAME(ClassSessions.Schedule) = 'Wednesday';

1. Request: Retrieve the names and contact details of clients who have cars with a mileage over 20,000 and are scheduled for a test this week:

SELECT Clients.Name, Clients.ContactDetails

FROM Clients

JOIN Cars ON Clients.InstructorID = Cars.InstructorID

JOIN TestSchedules ON Clients.Clientid = TestSchedules.Clientid

WHERE Cars.MileageCovered > 20000

AND WEEK(TestSchedules.Schedule) = WEEK(NOW());

1. Request: Find the average mileage covered by each type of car:

SELECT Type, AVG(MileageCovered) AS AverageMileage

FROM Cars

GROUP BY Type;

1. Request: Retrieve the names of instructors who have not conducted any class sessions:

SELECT Instructors.Name

FROM Instructors

LEFT JOIN ClassSessions ON Instructors.InstructorID = ClassSessions.InstructorID

WHERE ClassSessions.SessionID IS NULL;

**Step 6.1 Begin with the list of the tables that the entities and relationships from the ER diagram mapped to naturally. For each table on the list, identify functional dependencies and normalize the relation to BCNF. Then decide whether the resulting tables should be implemented in that form. If not, explain why.**

**Instructors** (InstructorId, Name, ClassSchedule, Address, *OfficeNumber*)

OfficeNumber is a foreign key which is referencing Office.

**Office** (OfficeNumber, Location, City, Zipcode, PhoneNumber)

**Administrators** (AdministratorId, Name, Address, PhoneNumber, *OfficeNumbe, ManagerId*)

ManagerId is a Foreign Key referencing Managers & OfficeNumber is a Foreign Key referencing Office.

**Managers** (ManagerId, Name, Address, PhoneNumber)

**ClassSessions** (SessionId, Schedule, *InstructorId*, RoomNumber)

InstructorId is a Foreign Key referencing Instructors.

**Cars** (PlateNumber, YearBuilt, Type, MileageCovered, *InstructorId*)

InstructorId is a Foreign Key that refer Instructors.

**Clients** (ClientId, Name, Address, ContactDetails, *InstructorId*)

InstructorId is a Foreign Key referencing Instructors  
**TestSchedules** (TestScheduleId, Schedule, *ClientId*)

ClientId is a Foreign Key referencing Clients.

**Mileage** (MileageId, Maintenance, EngineSize, WeightOfCar, FuelEfficiency, *PlateNumber*)

PlateNumber is a Foreign key referencing Cars.

**ApplicationForm** (FormId, Details, *ClientId*)

ClientId is a Foreign key which refer Client.

**Instructors:**

Functional Dependencies:

InstructorId → Name, ClassSchedule, Address, OfficeNumber

Normalize to BCNF:

Instructor (InstructorId, Name, ClassSchedule, Address, OfficeNumber)

**Office:**

Functional Dependencies:

OfficeNumber → Location, City, Zipcode, PhoneNumber

Normalize to BCNF:

Office (OfficeNumber, Location, City, Zipcode, PhoneNumber)

**Administrators:**

Functional Dependencies:

AdministratorId → Name, Address, PhoneNumber, ManagerId, OfficeNumber

Normalize to BCNF:

Administrator (AdministratorId, Name, Address, PhoneNumber, ManagerId, OfficeNumber)

**Managers:**

Functional Dependencies:

ManagerId → Name, Address, PhoneNumber

Normalize to BCNF:

Manager (ManagerId, Name, Address, PhoneNumber)

**ClassSessions:**

Functional Dependencies:

SessionId → Schedule, InstructorId, RoomNumber

Normalize to BCNF:

ClassSession (SessionId, Schedule, InstructorId, RoomNumber)

**Cars:**

Functional Dependencies:

PlateNumber → YearBuilt, Type, MileageCovered, Maintenance, EngineSize, WeightOfCar, FuelEfficiency, InstructorId

Normalize to BCNF:

Car (PlateNumber, YearBuilt, Type, MileageCovered, Maintenance, EngineSize, WeightOfCar, FuelEfficiency, InstructorId)

**Clients:**

Functional Dependencies:

ClientId → Name, Address, ContactDetails

Normalize to BCNF:

Client (ClientId, Name, Address, ContactDetails)

**TestSchedules:**

Functional Dependencies:

TestScheduleId → Schedule, ClientId

Normalize to BCNF:

TestSchedule (TestScheduleId, Schedule, ClientId)

**Mileage:**

Functional Dependencies:

MileageId → Maintenance, EngineSize, WeightOfCar, FuelEfficiency, PlateNumber

Normalize to BCNF:

Mileage (MileageId, Maintenance, EngineSize, WeightOfCar, FuelEfficiency, PlateNumber)

**ApplicationForm:**

Functional Dependencies:

FormId → Details, ClientId

Normalize to BCNF:

ApplicationForm (FormId, Details, ClientId)

The resulting tables are now in Boyce-Codd Normal Form (BCNF), which eliminates redundancy and ensures that each table represents a single, cohesive theme.

**Step 6.3 For each table, write the table name and write out the names, data types, and sizes of all the data items, identify any constraints, using the conventions of the DBMS you will use for implementation.**

Below is a representation of each table with their respective names, data types, and sizes, along with identified constraints.

Instructors:

Table Name: Instructor

Columns:

InstructorId (INT, Primary Key)

Name (VARCHAR(50))

ClassSchedule (VARCHAR(50))

Address (VARCHAR(100))

OfficeNumber (INT, Foreign Key referencing Office.OfficeNumber)

Office:

Table Name: Office

Columns:

OfficeNumber (INT, Primary Key)

Location (VARCHAR(100))

City (VARCHAR(50))

Zipcode (VARCHAR(10))

PhoneNumber (VARCHAR(15))

Administrators:

Table Name: Administrator

Columns:

AdministratorId (INT, Primary Key)

Name (VARCHAR(50))

Address (VARCHAR(100))

PhoneNumber (VARCHAR(15))

ManagerId (INT, Foreign Key referencing Managers.ManagerId)

OfficeNumber (INT, Foreign Key referencing Office.OfficeNumber)

Managers:

Table Name: Manager

Columns:

ManagerId (INT, Primary Key)

Name (VARCHAR(50))

Address (VARCHAR(100))

PhoneNumber (VARCHAR(15))

ClassSessions:

Table Name: ClassSession

Columns:

SessionId (INT, Primary Key)

Schedule (VARCHAR(50))

InstructorId (INT, Foreign Key referencing Instructors.InstructorId)

RoomNumber (VARCHAR(20))

Cars:

Table Name: Car

Columns:

PlateNumber (VARCHAR(15), Primary Key)

YearBuilt (INT)

Type (VARCHAR(50))

MileageCovered (INT)

Maintenance (VARCHAR(50))

EngineSize (VARCHAR(20))

WeightOfCar (INT)

FuelEfficiency (VARCHAR(20))

InstructorId (INT, Foreign Key referencing Instructors.InstructorId)

Clients:

Table Name: Client

Columns:

ClientId (INT, Primary Key)

Name (VARCHAR(50))

Address (VARCHAR(100))

ContactDetails (VARCHAR(100))

TestSchedules:

Table Name: TestSchedule

Columns:

TestScheduleId (INT, Primary Key)

Schedule (VARCHAR(50))

ClientId (INT, Foreign Key referencing Clients.ClientId)

Mileage:

Table Name: Mileage

Columns:

MileageId (INT, Primary Key)

Maintenance (VARCHAR(50))

EngineSize (VARCHAR(20))

WeightOfCar (INT)

FuelEfficiency (VARCHAR(20))

PlateNumber (VARCHAR(15), Foreign Key referencing Cars.PlateNumber)

ApplicationForm:

Table Name: ApplicationForm

Columns:

ApplicationId (INT, Primary Key)

Details (VARCHAR(200))

ClientId (INT, Foreign Key referencing Clients.ClientId)

**Step 6.4 Write and execute SQL statements to create all the tables needed to implement the design.**

CREATE TABLE Instructor (

InstructorId INT PRIMARY KEY,

Name VARCHAR(50),

ClassSchedule VARCHAR(50),

Address VARCHAR(100),

OfficeNumber INT,

FOREIGN KEY (OfficeNumber) REFERENCES Office(OfficeNumber)

);

CREATE TABLE Office (

OfficeNumber INT PRIMARY KEY,

Location VARCHAR(100),

City VARCHAR(50),

Zipcode VARCHAR(10),

PhoneNumber VARCHAR(15)

);

CREATE TABLE Administrator (

AdministratorId INT PRIMARY KEY,

Name VARCHAR(50),

Address VARCHAR(100),

PhoneNumber VARCHAR(15),

ManagerId INT,

OfficeNumber INT,

FOREIGN KEY (ManagerId) REFERENCES Manager(ManagerId),

FOREIGN KEY (OfficeNumber) REFERENCES Office(OfficeNumber)

);

CREATE TABLE Manager (

ManagerId INT PRIMARY KEY,

Name VARCHAR(50),

Address VARCHAR(100),

PhoneNumber VARCHAR(15)

);

CREATE TABLE ClassSession (

SessionId INT PRIMARY KEY,

Schedule VARCHAR(50),

InstructorId INT,

RoomNumber VARCHAR(20),

FOREIGN KEY (InstructorId) REFERENCES Instructor(InstructorId)

);

CREATE TABLE Car (

PlateNumber VARCHAR(15) PRIMARY KEY,

YearBuilt INT,

Type VARCHAR(50),

MileageCovered INT,

InstructorId INT,

FOREIGN KEY (InstructorId) REFERENCES Instructor(InstructorId)

);

CREATE TABLE Client (

ClientId INT PRIMARY KEY,

Name VARCHAR(50),

Address VARCHAR(100),

ContactDetails VARCHAR(100)

);

CREATE TABLE TestSchedule (

TestScheduleId INT PRIMARY KEY,

Schedule VARCHAR(50),

ClientId INT,

FOREIGN KEY (ClientId) REFERENCES Client(ClientId)

);

CREATE TABLE Mileage (

MileageId INT PRIMARY KEY,

Maintenance VARCHAR(50),

EngineSize VARCHAR(20),

WeightOfCar INT,

FuelEfficiency VARCHAR(20),

PlateNumber VARCHAR(15),

FOREIGN KEY (PlateNumber) REFERENCES Car(PlateNumber)

);

CREATE TABLE ApplicationForm (

ApplicationID INT PRIMARY KEY,

Details VARCHAR(200),

ClientId INT,

FOREIGN KEY (ClientId) REFERENCES Client(ClientId)

);

A screenshot of a computer

Description automatically generated

**Step 6.5 Create indexes for foreign keys and any other columns that will be used most often for queries.**

**Index for foreign key in Instructors table**

CREATE INDEX idx\_Instructors\_OfficeNumber ON Instructor (OfficeNumber);

**Index for foreign keys in Administrators table**

CREATE INDEX idx\_Administrators\_ManagerId ON Administrator (ManagerId);

CREATE INDEX idx\_Administrators\_OfficeNumber ON Administrator (OfficeNumber);

**Index for foreign key in ClassSessions table**

CREATE INDEX idx\_ClassSessions\_InstructorId ON ClassSession (InstructorId);

**index for foreign keys in Cars table**

CREATE INDEX idx\_Cars\_InstructorId ON Car (InstructorId);

**Index for foreign key in TestSchedules table**

CREATE INDEX idx\_TestSchedules\_ClientId ON TestSchedule (ClientId);

**Index for foreign key in Mileage table**

CREATE INDEX idx\_Mileage\_PlateNumber ON Mileage (PlateNumber);

**Index for foreign key in ApplicationForm table**

CREATE INDEX idx\_ApplicationForm\_ClientId ON ApplicationForm (ClientId)

**Step 6.6 Insert about five records in each table, preserving all constraints. Put in enough data to demonstrate how the database will function.**

INSERT INTO Instructor (InstructorId, Name, ClassSchedule, Address, OfficeNumber) VALUES

(1, 'John Doe', 'Monday 10:00 AM - 12:00 PM', '123 Main St', 101),

(2, 'Jane Smith', 'Wednesday 2:00 PM - 4:00 PM', '456 Oak St', 102),

(3, 'Bob Johnson', 'Tuesday 9:00 AM - 11:00 AM', '789 Pine St', 103),

(4, 'Alice Davis', 'Thursday 1:00 PM - 3:00 PM', '101 Elm St', 104),

(5, 'Charlie Brown', 'Friday 3:00 PM - 5:00 PM', '202 Maple St', 105);

INSERT INTO Office (OfficeNumber, Location, City, Zipcode, PhoneNumber) VALUES

(101, 'Building A', 'City1', '12345', '555-1111'),

(102, 'Building B', 'City2', '23456', '555-2222'),

(103, 'Building C', 'City3', '34567', '555-3333'),

(104, 'Building D', 'City4', '45678', '555-4444'),

(105, 'Building E', 'City5', '56789', '555-5555');

INSERT INTO Administrator (AdministratorId, Name, Address, PhoneNumber, ManagerId, OfficeNumber) VALUES

(1, 'Admin1', 'AdminAddress1', '111-1111', 1, 101),

(2, 'Admin2', 'AdminAddress2', '222-2222', 2, 102),

(3, 'Admin3', 'AdminAddress3', '333-3333', 3, 103),

(4, 'Admin4', 'AdminAddress4', '444-4444', 4, 104),

(5, 'Admin5', 'AdminAddress5', '555-5555', 5, 105);

INSERT INTO Manager (ManagerId, Name, Address, PhoneNumber) VALUES

(1, 'Manager1', 'ManagerAddress1', '111-1111'),

(2, 'Manager2', 'ManagerAddress2', '222-2222'),

(3, 'Manager3', 'ManagerAddress3', '333-3333'),

(4, 'Manager4', 'ManagerAddress4', '444-4444'),

(5, 'Manager5', 'ManagerAddress5', '555-5555');

INSERT INTO ClassSession (SessionId, Schedule, InstructorId, RoomNumber) VALUES

(1, 'Monday 10:00 AM - 12:00 PM', 1, 'Room101'),

(2, 'Wednesday 2:00 PM - 4:00 PM', 2, 'Room102'),

(3, 'Tuesday 9:00 AM - 11:00 AM', 3, 'Room103'),

(4, 'Thursday 1:00 PM - 3:00 PM', 4, 'Room104'),

(5, 'Friday 3:00 PM - 5:00 PM', 5, 'Room105');

INSERT INTO Car (PlateNumber, YearBuilt, Type, MileageCovered, InstructorId) VALUES

('ABC123', 2020, 'Sedan', 50000, 1),

('XYZ789', 2019, 'SUV', 60000, 2),

('DEF456', 2021, 'Hatchback', 40000, 3),

('GHI789', 2018, 'Coupe', 70000, 4),

('JKL012', 2022, 'Convertible', 30000, 5);

INSERT INTO Client (ClientId, Name, Address, ContactDetails) VALUES

(1, 'Client1', 'ClientAddress1', '111-1111'),

(2, 'Client2', 'ClientAddress2', '222-2222'),

(3, 'Client3', 'ClientAddress3', '333-3333'),

(4, 'Client4', 'ClientAddress4', '444-4444'),

(5, 'Client5', 'ClientAddress5', '555-5555');

INSERT INTO TestSchedule (TestScheduleId, Schedule, ClientId) VALUES

(1, 'Monday 1:00 PM', 1),

(2, 'Wednesday 3:00 PM', 2),

(3, 'Tuesday 2:00 PM', 3),

(4, 'Thursday 4:00 PM', 4),

(5, 'Friday 5:00 PM', 5);

INSERT INTO Mileage (MileageId, Maintenance, EngineSize, WeightOfCar, FuelEfficiency, PlateNumber) VALUES

(1, 'Regular Checkup', '2.0L', 1500, '30 MPG', 'ABC123'),

(2, 'Premium Service', '3.0L', 2000, '25 MPG', 'XYZ789'),

(3, 'Regular Maintenance', '1.5L', 1200, '35 MPG', 'DEF456'),

(4, 'Major Repair', '2.5L', 1800, '28 MPG', 'GHI789'),

(5, 'Oil Change', '1.8L', 1400, '32 MPG', 'JKL012');

INSERT INTO ApplicationForm (ApplicationId, Details, DateSubmitted, Status, ClientId) VALUES

(1, 'Details for Form 1', 1),

(2, 'Details for Form 2', 2),

(3, 'Details for Form 3', 3),

(4, 'Details for Form 4', 4),

(5, 'Details for Form 5', 5);

**Step 6.7 Write SQL statements that will process five non-routine requests for information from the database just created. For each, write the request in English, followed by the corresponding SQL command.**

Request 1: Retrieve the names and contact details of clients who have a scheduled test on Wednesday.

SELECT c.Name, c.ContactDetails

FROM Clients c

JOIN TestSchedules ts ON c.ClientId = ts.ClientId

JOIN ClassSessions cs ON ts.Schedule = cs.Schedule

WHERE cs.Schedule LIKE 'Wednesday%';

Request 2: Find instructors who have cars with fuel efficiency greater than 30 MPG.

SELECT DISTINCT i.Name

FROM Instructors i JOIN Cars c

ON i.InstructorId = c.InstructorId

WHERE c.mileagecovered > '30 MPG';

Request 3: List the offices and the total number of administrators in each office.

SELECT o.OfficeNumber, COUNT(a.AdministratorId) AS NumberOfAdministrators

FROM Office o LEFT JOIN Administrators a

ON o.OfficeNumber = a.OfficeNumber

GROUP BY o.OfficeNumber;

Request 4: Retrieve the details of application forms submitted by clients who have scheduled tests.

SELECT af.ApplicationId, af.Details, af.ClientId, ts.Schedule

FROM ApplicationForm af

JOIN TestSchedules ts ON af.ClientId = ts.ClientId;

Request 5: Find the average mileage covered by cars for each type of maintenance.

SELECT m.Maintenance, AVG(MileageCovered) AS AverageMileage

FROM Mileage m

join cars c on m.PlateNumber = c.PlateNumber

GROUP BY Maintenance;

**##Learnings from this project so far:**

**Data Modeling and Database Design**: In order to create an efficient and successful database management system, the project stressed the significance of careful data modeling and database design. Determining the relationships, entities, and characteristics is essential to a successful database deployment.

**Entity-Relationship Modeling**: Understanding the relationships between different entities, such as offices, administrators, instructors, clients, and more, is essential. It ensures the database accurately reflects the real-world connections between different aspects of the business.

**Normalization**: Learning the principles of normalization to eliminate data redundancy and improve data integrity. Properly normalizing the database schema helps avoid anomalies and ensures data consistency.

**Foreign Key Relationships**: Implementing foreign key relationships between tables to establish connections and maintain referential integrity. This is crucial for maintaining consistency across the database.

**##Challenges faced from this project so far:**

**Complexity of Relationships**: Managing relationships between various entities, especially in a business with multiple offices and diverse interactions, can be challenging. Ensuring that each relationship is correctly represented and maintained requires careful consideration.

**Data Consistency**: Ensuring data consistency across different tables and entities can be challenging, especially when dealing with updates, inserts, and deletions. Implementing transactions and constraints effectively helps mitigate this challenge.

**##Conclusion:**

**Efficient Data Management**: The new database system will significantly improve data management for Smooth Drive College of Motoring. It addresses communication issues, streamlines operations, and provides a centralized repository for information.

**Enhanced Services**: With a well-designed database system, the school can enhance its services by efficiently managing client information, instructor records, lesson schedules, and more. This contributes to a better overall experience for clients and staff.

**Continuous Improvement**: The conclusion highlights the importance of continuous improvement. Regular updates and optimizations to the database system based on user feedback and changing business requirements are crucial for long-term success.

**Training and Adoption**: Implementing the new system will require proper training for staff to ensure a smooth transition. User adoption is a critical factor in the success of any database management system.

In summary, the project provides valuable insights into the complexities and rewards of creating a comprehensive database system. It reinforces the importance of aligning the database design with real-world business processes and the need for ongoing improvements to meet evolving needs.