**F28DM**

**Database Management Systems**

**Group 14**

**Ziyad Alrasbi, Ben Gardner,**

**Hasan Kapadia, Mohaamed Ashab Uddin**

**Heriot-Watt University Edinburgh**

Contents

[T1-E.R Diagram](#_T1-_E.R_Diagram)----------------------------------------------------------------------2

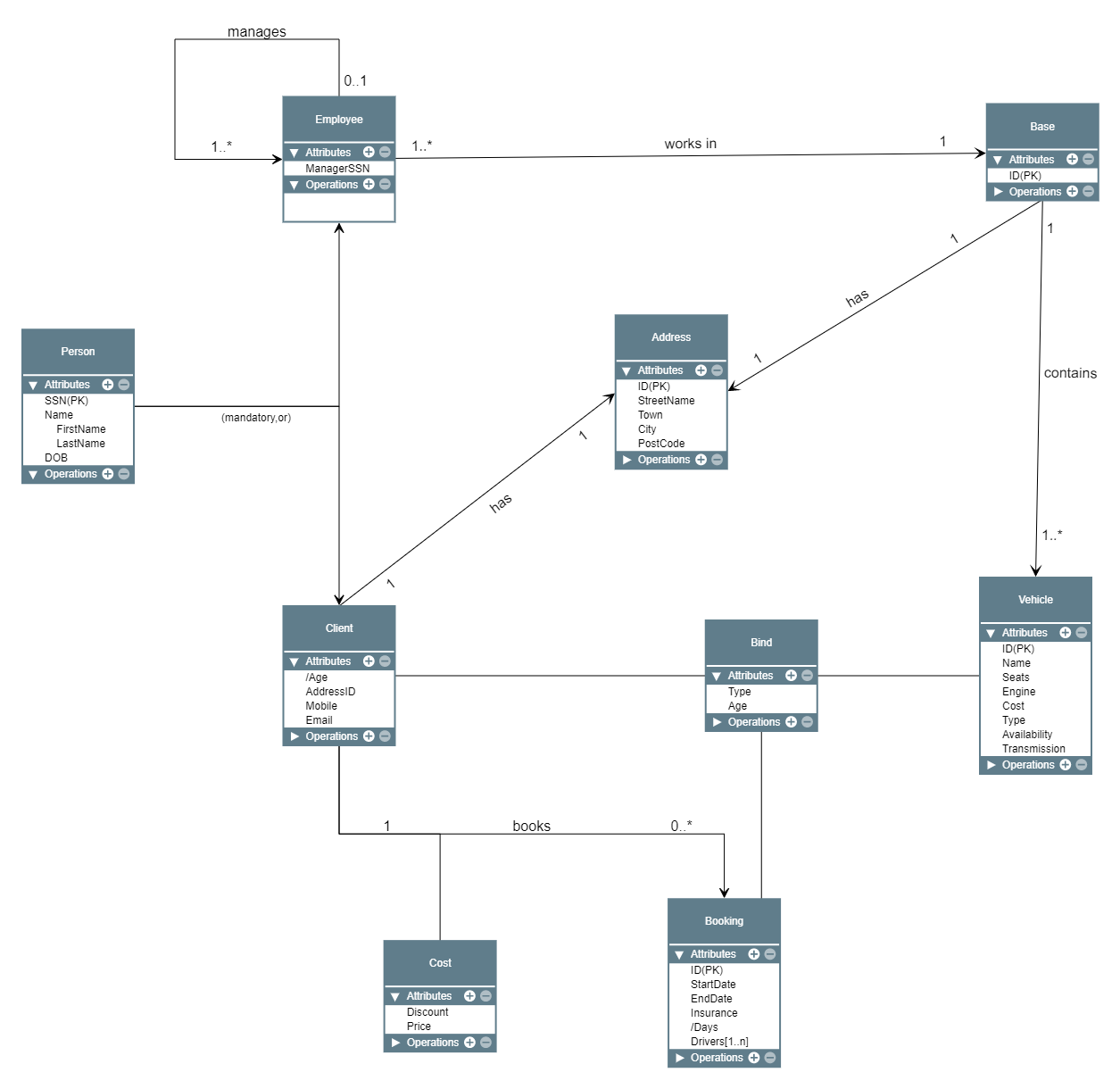
[T2- Relational Schema](#_T2-_Relational_Schema)-------------------------------------------------------------3

[T3- MySQL Schema](#_T3-_MySQL_Schema)-----------------------------------------------------------------5

[T4- Indexes](#_T4_–_Indexes)---------------------------------------------------------------------------8

[Links](#_Links)------------------------------------------------------------------------------------9

# T1- E.R Diagram



Comments**-**

* Bind entity used to check whether the client can only take a specific vehicle type or any vehicle type based on their age
* Cost entity is used to calculate the cost of the vehicle after discounts are applied
* In the client entity ‘age’ is a derived value from DOB and in Booking ‘days’ is a derived value from the difference between the StartDate and the EndDate

# T2- Relational Schema

Employee (**SSN**, FirstName, LastName, DOB, *BaseID*, ManagerSSN)

Booking (**BookID**, SSN, StartDate, EndDate, Insurance, /Day, *VehicleID*)

Client (**SSN**, FirstName, LastName, DOB, /Age, *AddressID*, Mobile, Email)

Bind (*ClientID*, *VehicleID*, Type, Age)

Drivers (**SSN**, **BookID**)

Address (**ID**, StreetName, Town, City, PostCode)

Vehicle (Name, **ID**, Seats, Engine, Cost, Type, Transmission, Availability, *BaseID*)

Base (**BaseID**, *AddressID*)

Cost (Price, Discount, *BookID*)

# T3- MySQL Schema

CREATE TABLE Address (

  ID BIGINT PRIMARY KEY AUTO\_INCREMENT,

  StreetName VARCHAR(50) NOT NULL,

  Town VARCHAR(20) NOT NULL,

  City VARCHAR(20) NOT NULL,

  PostCode VARCHAR(12) NOT NULL

) ENGINE=INNODB;

CREATE TABLE Client (

  SSN VARCHAR(20) PRIMARY KEY NOT NULL,

  LicenseID VARCHAR(20) UNIQUE,

  FirstName VARCHAR(20) NOT NULL,

  LastName VARCHAR(20) NOT NULL,

  DOB DATE NOT NULL,

  Age TINYINT CHECK (Age>=25),

  AddressID BIGINT,

  FOREIGN KEY (AddressID) REFERENCES Address(ID),

  Mobile CHAR(10) UNIQUE,

  Email VARCHAR(50) UNIQUE

) ENGINE=INNODB;

CREATE TABLE Base (

  ID INT PRIMARY KEY AUTO\_INCREMENT,

  AddressID BIGINT,

  FOREIGN KEY (AddressID) REFERENCES Address(ID)

) ENGINE=INNODB;

CREATE TABLE Employee (

  SSN VARCHAR(20) PRIMARY KEY,

  FirstName VARCHAR(20) NOT NULL,

  LastName VARCHAR(20) NOT NULL,

  DOB DATE NOT NULL,

  BaseID INT,

  FOREIGN KEY (BaseID) REFERENCES Base(ID),

  ManagerSSN VARCHAR(20)

) ENGINE=INNODB;

CREATE TABLE Vehicle (

  ID BIGINT PRIMARY KEY AUTO\_INCREMENT,

  Name VARCHAR(50) NOT NULL,

  Seats TINYINT DEFAULT 4,

  Engine VARCHAR(20) NOT NULL,

  Cost TINYINT(100) NOT NULL,

  Type VARCHAR(50) NOT NULL CHECK (Type='SmallTownCar' OR Type='FamilyCar' OR Type='MPV'

 OR Type='SportsCar' OR Type='Luxury' OR Type='Minivans'),

  Availability BOOLEAN,

  Transmission VARCHAR(50) DEFAULT 'Automatic' CHECK (Transmission='Automatic' OR

Transmission='Manual'),

  BaseID INT,

  FOREIGN KEY (BaseID) REFERENCES Base(ID)

) ENGINE=INNODB;

CREATE TABLE Booking (

  ID INT PRIMARY KEY AUTO\_INCREMENT,

  StartDate DATE NOT NULL,

  EndDate DATE NOT NULL,

  Insurance BOOLEAN DEFAULT False,

  Day TINYINT DEFAULT 1,

  VehicleID BIGINT,

  SSN VARCHAR(20),

  FOREIGN KEY (SSN) REFERENCES Client(SSN),

  FOREIGN KEY (VehicleID) REFERENCES Vehicle(ID)

) ENGINE=INNODB;

CREATE TABLE Bind (

  ClientID VARCHAR(20),

  VehicleID BIGINT,

  FOREIGN KEY (VehicleID) REFERENCES Vehicle(ID),

  PRIMARY KEY (ClientID),

  FOREIGN KEY (ClientID) REFERENCES Client(SSN),

  Type VARCHAR(50),

  Age TINYINT

) ENGINE=INNODB;

CREATE TABLE Cost (

  Discount INT,

  Price INT,

  BookID INT,

  FOREIGN KEY (BookID) REFERENCES Booking(ID),

  Primary KEY(BookID, Price)

) ENGINE=INNODB;

CREATE TABLE Drivers (

  SSN VARCHAR(20),

  BookID INT,

  FOREIGN KEY (SSN) REFERENCES Client(SSN),

  FOREIGN KEY (BookID) REFERENCES Booking(ID),

  Primary KEY(BookID, SSN)

) ENGINE=INNODB;

/\*

\*\*Employee\*\*

Employee class is created to allow us to have recursive relation.

A manager supervises an employee, who is also an employee, which

makes up the recursive relation. It has many to one relationship

with base.

\*\*Base\*\*

Base has one to one relationship with Address Entity.

\*\*Vehicle\*\*

Vehicle has its necessary specifications and many to one

relationship with Base entity.

\*\*Booking\*\*

This entity holds booking details from any booking made by a client.

Therefore, it has many to one relationship. There is also an

Associated class (Cost) that decides the cost of the booking after discount

In the booking to client relation. It has the references to the vehicle with

Many to one relation.

The Day attribute is a derived attribute calculated from the StartDate & Enddate attributes

Which is calculated with (EndDate - StartDate)

The price and discount columns are calculated using the cost column from the vehicle table and

The day’s column from the booking table,

Which calculates the cost of booking the vehcle for the required amount of days

But if the client books the vehicle for > 7 days a discount of 20% will be applicable and

Will be automatically cut from the price.

\*\*Address\*\*

Stores the address of clients and bases, has a one-to-one relationship with client

And base tables.

\*\*Drivers\*\*

This the entity that holds the additional drivers from the booking.

Therefore, it has many to one relationship with the booking.

\*\*Client\*\*

It holds the information of a client. Has a one to one relation with address

And one to many relation with booking

\*\*Bind\*\*

This class will determine the type of cars that a person can have according

To their age restriction.

Age works around a check function which uses the datediff() function to get the current date with

getdate()

And the date given in the DOB column in Client to get the age of the driver.

This in turn allows us to check and apply a constraint stating that if the age of the client is above 25

But below 30 the car types they are allowed to take is restricted whereas if there age is above 30

They can take any car type.

\*/

# T4 – Indexes

CREATE INDEX EmployeeID ON Employee(FirstName, LastName, BaseID);

CREATE INDEX BookingDetails ON Booking(SSN, VehicleID, Day);

CREATE INDEX VehicleID ON Vehicle(Name, Type, Availability, Cost);

CREATE INDEX ClientDetails ON Client(FirstName,LastName,Age,Mobile,Email);

CREATE INDEX Address ON Address(StreetName, City, PostCode);

/\*

\*\*EmployeeID\*\*

As the company HW Motors has many bases across UK and are a company focused on expanding

The company would have a number of employees with similar first names and last names

An index would be useful on the employee's (FirstName, LastName, BaseID) to make it

Easier to find the employees in the company.

\*\*BookingDetails\*\*

HWMotors as an expanding comnpany would have a lot of Bookings with the same client

So to narrow it down and make it easier to search having an index on (SSN, VehicleID, Day)

Would make it easier to find the specific booking

\*\*VehicleID\*\*

Would make it easier to find if a specific vehicle is available and if the vehicle type is

Suited for the client.

\*\*ClientDetails\*\*

Similar to EmployeeID as the company is an expanding company with good aims the company

Is bound

To have many clients with similar names and DOBs. An index on Client would provide the

Ability to

Search for the details of a client more efficiently and faster.

\*\*Address\*\*

Address is an entity which contains the address for all the companies’ clients and bases

As these

Attributes can have a multitude of enteries having an index here could make it easier to

Search or a specific address using (StreetName, City, PostCode).

\*/

# Links

GitHub Repo (Has all work done)-

<https://github.com/HasanKaps/DBMSCW>

E.R Diagram-

<https://www.yworks.com/yed-live/?file=https://gist.githubusercontent.com/HasanKaps/49b49d9f5c81fd543d85f19152ac7f61/raw/HWMotors>