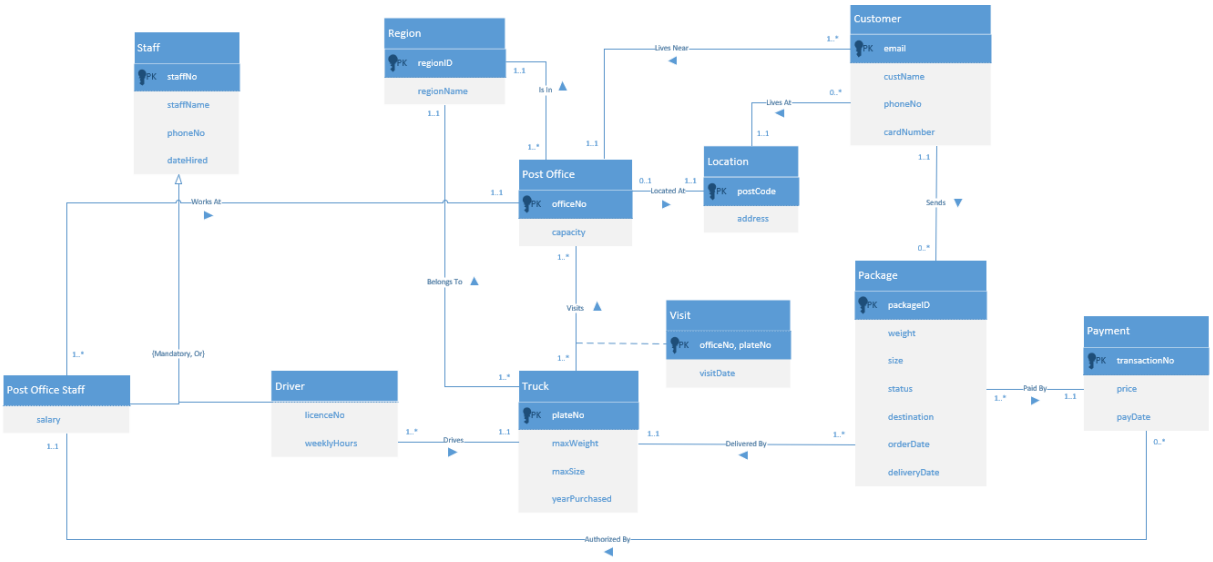
1. ER Diagram



2. Relational Database Schema

Customer(email, custName, postCode, phoneNo, cardNo, officeNo)
Primary Key: email
Foreign Key: officeNo references PostOffice(officeNo)
Foreign Key: postCode references Location(postCode)

Location(postCode, address)
Primary Key: postCode

Package(packageID, weight, size, destination, orderDate, deliveryDate, senderEmail, transactionNo, plateNo)
Primary Key: packageID
Foreign Key: senderEmail references Customer(email)
Foreign Key: transactionNo references Payment(transactionNo)
Foreign Key: plateNo references Truck(plateNo)

Payment(transactionNo, price, payDate, packageID)
Primary Key: transactionNo
Foreign Key: packageID references Package(packageID)

PostOffice(officeNo, postCode, capacity, regionID)
Primary Key: officeNo
Foreign Key: regionID references Region(regionID)
Foreign Key: postCode references Location(postCode)

PostOfficeStaff(staffNo, staffName, phoneNo, dateHired, salary, officeNo)
Primary Key: staffNo
Foreign Key: officeNo references PostOffice(officeNo)

Driver(staffNo, staffName, phoneNo, dateHired, licenseNo, weeklyHours, plateNo)
Primary Key: staffNo
Foreign Key: plateNoNo references Truck(plateNo)

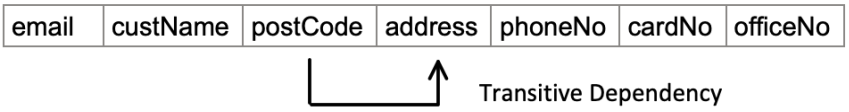
Truck(plateNo, maxWeight, maxSize, yearPurchased, regionID)
Primary Key: plateNo
Foreign Key: regionID references Region(regionID)

Visit(officeNo, plateNo, visitDate)
Primary Key: officeNo, plateNo
Foreign Key: officeNo references PostOffice(officeNo)
Foreign Key: plateNo references Truck(plateNo)

Region(regionID, regionName)
Primary Key: regionID

3. **Schema Revision**
    a. **Normalization**

Customer:



The ‘address’ attribute is transitively dependent on the ‘postCode’ attribute. We will make a separate relation called *Location* that contains just these attributes.

Package:

The “officeNo” attribute is transitively dependent on the “senderEmail” attribute, which is a foreign key referencing the *Customer* relation. Due to the transitive properties of the tables, you are able to get the office attribute “officeNo” for each package through the foreign key:

senderEmail references Customer(email)

and then in the *Customer* relation:

officeNo references PostOffice(officeNo)

Payment:

transactionNo	price	payDate	packageID	senderEmail

The “senderEmail” attribute is transitively dependent on the “packageID” attribute, which is a foreign key referencing the *Package* relation. Due to the transitive properties of the tables, you are able to get the payment attribute “senderEmail” for each payment using the foreign key:

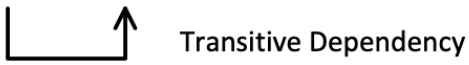
packageID references Package(packageID)

and then in the *Package* relation :

senderEmail references Customer(email)

PostOffice:

officeNo	postCode	address	capacity	regionID



The ‘address’ attribute is transitively dependent on the ‘postCode’ attribute. We will make a separate relation called *Location* that contains just these attributes. (Same relation that we created when normalizing *Customer*)

PostOfficeStaff:

staffNo	staffName	phoneNo	dateHired	salary	officeNo

There were no changes to this table, the relation is in 3rd normal form.

Driver:

staffNo	staffName	phoneNo	dateHired	licenseNo	weeklyHours	plateNo

There were no changes to this table, the relation is in 3rd normal form.

Truck:

plateNo	maxWeight	maxSize	yearPurchased	regionID

There were no changes to this table, the relation is in 3rd normal form.

Visit:

officeNo	plateNo	visitDate

There were no changes to this table, the relation is in 3rd normal form.

Region:

regionID	regionName

There were no changes to this table, the relation is in 3rd normal form.

b. Combining Relations

There are no opportunities to combine relations without introducing redundancy

c. Non-BCNF Schemas

There are no examples of non-BCNF relation schemas.