# Database Normalisation

## Customer

### Un-normalised

**Customer**( CustomerID, Name, StreetNumber, streetName, suburb, state, Postcode, Email, Phone Number, Password)

### 1st Normal Form

Some attributes are not atomic values that should be such as Name.

**Customer**( CustomerID, fName, lName StreetNumber, streetName, suburb, state, Postcode, Email, Phone Number, Password)

### 2nd Normal Form

Most attributes are functionally dependent on the Customer ID however the Address within itself would be a composite Key and will therefore become its own class, thus to restore the atomic values. This shall also be done to create a single separate class for login.

**Customer**( CustomerID, fName, lName, rAddress, bAddress, Email, Phone Number)

**Address(** addressID, StreetNum, streetName, suburb, State, PostCode)

**CustomerLogin(**customerPassword, CustomerID, CustomerEmail)

### 3rd Normal Form

This form is not needed as all information is dependent on the Primary key. And as such the risk of any anomalies or conflicts has been severely lessened.

### Fully Normalized

**Customer**(CustomerID, fName, lName, rAddress, bAddress, customerEmail, customerPhoneNum)

**PRIMARY KEY** CustomerID

**FOREIGN KEY** rAddress REFERENCES Address(AddressID) ON DELETE NO ACTION

**FOREIGN KEY** bAddress REFERENCES Address(AddressID) ON DELETE NO ACTION

**Address(**addressID, StreetNum, streetName, suburb, State, PostCode)

**PRIMARY KEY** AddressID

**CustomerLogin(**customerPassword, CustomerID, CustomerEmail)

**FOREIGN KEY** CustomerID References Customer(CustomerID)

## Admin

### Un-Normalised

**Admin(**AdminID, Name, Email, password)

### 1st Normal Form

This went through the same sort of process as the 1st Normal Form and as such will be essentially the same sort of deal

**Admin(** AdminID, fname, lName, Email, Password)

### 2nd Normal Form

Login to ensure that it is its own weak but connected entity will be made a separate table

**Admin(** adminID, fName, lName, adminEmail)

**AdminLogin**( adminID, adminEmail, adminPassword)

### 3rd Normal Form

Tables already in 3rd Normal Form

### Normalised

**Admin(** adminID, fName, lName, adminEmail)

**PRIMARY KEY** adminID

**AdminLogin**( adminID, adminEmail, adminPassword)

**FOREIGN KEY** adminID REFERENCES Admin(AdminID) ON UPDATE NO ACTION ON DELETE RESTRICT SET NULL

## Product

This table required veryu little normalisation as all the values are already atomic and everything is functionally dependent on the primary Key. The only major Normalisation of this and the next table will be shown in Order.

**Product**( ProductID, prodSize, prodPrice, shortDesc, longDesc, prodGender, imageFile, prodStock, lastEdited, active)

**PRIMARY KEY** ProductID

**FOREIGN KEY** lastEdited **REFERENCES** Admin(AdminID)

## Order

### Un-Normalised

**Order**( orderID, orderDate, subtotal, customerID, customerAddress, ship Method, tax, orderTotal, ProductID, Quantity, Price, cardType, CardNo, expirationMonth, Expirationyear)

### Normalisation:

Due to the unatomic nature of certain variables as well as their functional dependency on a foreign key (Product ID). Due to this being a many to many dependencies. This is fixed by including a junction box between them.

**Order(** orderID, orderDate, subtotal, customerID, customerAddress, ship Method, tax, orderTotal, cardType, CardNo, expirationMonth, Expirationyear)

**PRIMARY KEY** orderID

**FOREIGN KEY** (customerID) REFERENCES tblCustomer(customerID)

**JunctionProd\_Order**( orderID,productid, prodPrice, Quantity)

**FOREIGN KEY** orderID REFERENCES Order(orderID),

**FOREIGN KEY** productID REFERENCES ProductproductID,

**CONSTRAINT** PK\_PRod\_Order PRIMARY KEY(

orderID,

productID

)