Additional Normalization Problem

Updated

Decomposition Into 3NF

Step 1:

A single relation for all of our attributes:

Orders (orderID, orderDate, orderTime, orderAddress, orderNotes, isPaid, isDelivered, customerID, CCNumber, foodID, quantity, restaurantID, customerPhoneNum, customerName, customerAddress, foodPrice, deilveryFee, tip, coupon, expDate, nameOnCard, foodName, restaurantAddress, restaurantPhoneNum, restaurantBankAccountNumber, restaurantRating, restaurantComment, driverID, driverName, driverPhoneNum, driverComment, driverRating, driverBankAccountNumber, D.accountType, D.bankName, D.routingNumber, R.accountType, R.bankName, R.routingNumber)

All the FDs:

orderID -> customerID, restaurantID, driverID,

orderDate, orderTime, orderAddress, orderNotes, isPaid, isDelivered,

deliveryFee, tip, coupon, CCNumber,

restaurantRating, restaurantComment, driverRating, driverComment

customerID -> customerPhoneNum, customerName, customerAddress

restaurantID -> restaurantAddress, restaurantPhoneNum, restaurantBankAccountNumber

driverID -> driverName, driverPhoneNum, driverBankAccountNumber

foodID -> restaurantID, foodName, foodPrice

orderID, foodID -> quantity

CCNumber -> expDate, nameOnCard

driverBankAccountNumber -> D.accountType, D.bankName, D.routingNumber

restaurantBankAccountNumber -> R.accountType, R.bankName, R.routingNumber

Step 2: Find minimal basis

Step 3: Use all LHS and RHS in a FD from minimal basis to form a relation

R1 Orders (orderID, customerID, restaurantID, driverID,

orderDate, orderTime, orderNotes, isPaid, isDelivered,

deliveryFee, tip, coupon, CCNumber,

restaurantRating, restaurantComment, driverRating, driverComment)

R2 Customer (customerID, customerPhoneNum, customerName, customerAddress)

R3 Restaurant (restaurantID, restaurantAddress, restaurantPhoneNum, restaurantBankAccountNumber)

R4 Driver (driverID, driverName, driverPhoneNum, driverBankAccountNumber)

R5 Food (foodID, restaurantID, foodName, foodPrice)

R6 FoodInOrder (foodID, orderID, quantity)

R7 CreditCard (CCNumber, expDate, nameOnCard)

R8 DriverBankAccount (driverBankAccountNumber, D.accountType, D.bankName, D.routingNumber)

R9 RestaurantBankAccount (restaurantBankAccountNumber, R.accountType, R.bankName, R.routingNumber)

Step 4: Check key:

|  |  |  |
| --- | --- | --- |
| L | M | R |
| orderID  foodID | customerID  driverID  restaurantID  CCNumber  driverBankAccountNumber  restaurantBankAccountNumber | orderDate, orderTime, orderAddress, orderNotes, isPaid, isDelivered, deliveryFee, tip, coupon, expDate, nameOnCard, restaurantRating, restaurantComment, driverRating, driverComment  customerPhoneNum, customerName, customerAddress  restaurantAddress, restaurantPhoneNum  foodName, foodPrice  Quantity, D.accountType, D.bankName, D.routingNumber, R.accountType, R.bankName, R.routingNumber,  expDate, nameOnCard |

The key is (orderID, foodID), which is already in FoodInOrder relation. So no additional relations are needed.

Chase Test for this design is in file: *Group1\_Chase\_Test\_Food\_Delivery*