**Problem 10. Normalize the table to 1NF according to the business rules stated below.**

The following table represents library book check-out information. Each transaction corresponds to a book-check-out transaction in real life, that is, each student may charge out many books in one transaction and each student may come to charge out books **many times during the same day**. However, each transaction has a unique tran#.

**Transaction: (tran#, checkout\_datetime, sid, sname, due\_date, CallNums, Titles)**

1. Each student has a unique sid, i.e., each sid corresponds to only one student. Each student has one name.
2. Each transaction (represented by tran#) can only be conducted by one student, can only occur on one checkout date/time.
3. All books charged out on the same date all have the same due date.
4. Call numbers uniquely identify each book. A book has one title.

**Transaction: (tran#, checkout\_datetime, sid, sname, due\_date, CallNums, Titles)**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Tran# | Checkout\_dateTime | Sid | Sname | Due\_date | CallNums | Titles |
| 1001 | 9/1/2016 2:00 pm | 1234 | Victor | 11/1/2016 | QA 232  QA555 | Optimization  Linear programming |
| 1002 | 9/8/2016 11:00 am | 1234 | Victor | 11/8/2016 | QA 76 | Database |

**Solution:**

**PK** – Tran# + CallNum

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Tran# | Checkout\_dateTime | Sid | Sname | Due\_date | CallNums | Titles |
| 1001 | 9/1/2016 2:00 pm | 1234 | Victor | 11/1/2016 | QA 232 | Optimization |
| 1001 | 9/1/2016 2:00 pm | 1234 | Victor | 11/1/2016 | QA555 | Linear programming |
| 1002 | 9/8/2016 11:00 am | 1234 | Victor | 11/8/2016 | QA 76 | Database |

**Problem 1**. **Answer Yes or No** to the following Functional Dependency questions. Base you answers on the data shown in Table LifeStory. There are no more rows in this table.

LifeStory:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| A | B | C | D | E | F |
| short | red | 1 | morning | hot | spring |
| short | red | 2 | afternoon | cold | summer |
| long | blue | 3 | evening | warm | fall |
| long | blue | 3 | evening | warm | spring |
| long | blue | 3 | night | warm | summer |

a). Does C→A? **Yes,** C has unique value over A

b). Does A→C? **No,** Short on column A has multiple value in C – 1,2

c). Does B+C→D? **No,** Combination of Blue and 3 has multiple values in D as evening and night

d). Does B+C→A? **Yes,** B+C is unique and have one value in A

e). Does A+B+C→E? **Yes,** C is unique and combination of A+B+C will become unique and have one value in E

**Problem** 3: Techno Realty is a Connecticut realty company. It employs 20 real estate agents working in five different offices. An agent works in only one office.

Techno Realty is very technology savvy. It is in the process of designing a central database. The database should store information about its offices and agents. When a property is listed for sale through Techno Realty, the property owner would work with one and only one real estate agent. The database should record the date of listing and the listing price. If the price of the property is reduced later on, the dates of reduction and the new lowered prices should also be recorded. When a potential buyer submits a bid for a property that Techno Realty lists, the database should record information about the bidder, the date of bid, bidding price, proposed closing date, and mortgage term (such as 20/80). The final winning bid is also recorded. Note that when an initial bid was rejected by the property owner, the bidder can revise his/her bid by submitting another bid. When the sale is closed, the closing date, price, and mortgage term are recorded.

To make things simple, we assume that each property is owned by one owner and each potential buyer is also a single-person buyer.

Answer **Yes or No** to the following questions (all attributes present in these questions are actual attributes in the final tables.)

1). Does PropertyId→ListingPrice? **No,** Price of the property (PropertyId) may change.

2). Does OfficeID→AgentId? **No,** one office may have multiple agents.

3). Does AgentId→OfficeId? **Yes**, every agent will be mapped to an office.

4). Does buyerId→bidPrice? **No**, Buyer can submit multiple bid price for the property.

5). Does PropertyId→BidPrice? **No,** Property may have multiple bid price.

**Problem 4: Identify functional dependencies as implied by the stated business rules.**

The following table records forum posts.

1. Each user has a unique user id. That user id corresponds to one email address.
2. Table records the date the user account was created.
3. Table records a user’s activity level for each activity type. (For example, Activity type 1 may be starting a new thread and user 100234 started 1 thread. Activity 2 may be posting in an existing thread and 100234 has 5 posts. Activity 3 may be replying to an existing post and 100234 replied 10 times, etc.)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| UserId | MembershipStarts | EmailAddress | Activity Type | Activity NumberOfTimes |
| 100234 | 1/3/2016 | [abc@gmail.com](mailto:abc@gmail.com) | 1 | 1 |
| 100234 | 1/3/2016 | [abc@gmail.com](mailto:abc@gmail.com) | 2 | 5 |
| 100234 | 1/3/2016 | [abc@gmail.com](mailto:abc@gmail.com) | 3 | 10 |
| 100234 | 1/3/2016 | [abc@gmail.com](mailto:abc@gmail.com) | 4 | 20 |
| 100345 | 2/5/2017 | [george12@yahoo.com](mailto:george12@yahoo.com) | 1 | 10 |
| 100345 | 2/5/2017 | [george12@yahoo.com](mailto:george12@yahoo.com) | 2 | 65 |
| 100345 | 2/5/2017 | [george12@yahoo.com](mailto:george12@yahoo.com) | 3 | 35 |

1. UserId 🡪 EmailAddress, MembershipStarts. Each UserId has unique Email Address and Membership Start date.
2. UserId+ActivityType 🡪 ActivityNumberOfTimes, combination of each User Id and Active type has a unique Activity Number of Times because it counts number of times user performs the activity of this type.

**Problem 12b**

**b). Complaint:** (Complaint#, ComplaintReason, CustID, CustName, CustAddress, ReviewDept)

Complaint# 🡪 ComplaintReason

CustID 🡪 CustName, CustAddress

ComplaintReason 🡪 ReviewDept

Step 1. Add LHS of functional dependency.

🡪 Complaint# + CustID + ComplaintReason

Step 2: Reducing remove the column in LHS if it appears on RHS of functional dependency.

* Complaint# + CustID

Step 3: Add if any column did not appear anywhere.

* Complaint# + CustID

**PK = Complaint# + CustID**

**Problem 12C**

**c). Complaint:** (Complaint#, ComplaintReason, CustID)

Complaint# 🡪 ComplaintReason

Step 1. Add LHS of functional dependency.

🡪 Complaint#

Step 2: Reducing remove the column in LHS if it appears on RHS of functional dependency.

* Nothing to remove. Complaint#

Step 3: Add if any column did not appear anywhere.

* Complaint# + CustID

**PK = Complaint# + CustID**

**Problem 13**: Normalize to 2NF then to BCNF (Simple)

a). CarServiceHistory: (Tran#, date, serviceType, labor\_charge, part\_charge, total\_charge)

Additional FD: Tran#→date, total\_charge

ServiceType→Labor\_charge, Part\_charge

Since Tran# and ServiceType is part of primary key, it violates the 2NF.

To normalize the table, remove the ServiceType from the table.

T1: (Tran#, serviceType)

T2: (Tran# date, total\_charge)

T3: (serviceType, labor\_charge, part\_charge)

T1,T2,T3 are all in BCNF.

b). Sale: (Date, Customer#, Product#, Vendor#, Vendor-City, SalesRepName)

Additional FD: Customer#→ SalesRepName;

Vendor#→Vendor-City

Normalization: Removing the Vendor# 🡪 Vendor-City from the table.

T1: (Date, Customer#, Product#, Vendor#, Vendor-City)

T2: (Customer#, SalesRepName)

For BCNF: we need to remove, Vendor#, Vendor-City and final normalized result is.

T1: (Date, Customer#, Product#, Vendor#)

T2: (Vendor#, Vendor-City)

T3: (Customer#, SalesRepName)

**Problem 14**: Normalize to 2NF then to BCNF (Advanced)

b). CarInspection: (Plate#, DateOfInspection model, year, owner-ssn, owner-phone, own-address, InspectionResult), where InspectionResult takes a value of either pass or fail,

Additional FD's: Plate#→ model, year, owner-ssn

owner-ssn→owner-address, owner-phone

Normalization:

T1: (Plate#, DateOfInspection, InspectionResult), and

T2 (Plate#, model, year, owner-ssn, owner-address, owner-phone)

Since T2 has owner-ssn, owner-address, owner-phone it violates the BCNF, to normalize.

T21: (Plate#, model, year, owner-ssn)

T22: (owner-ssn, owner-address, owner-phone)

Final Tables are

1. (Plate#, model, year, owner-ssn)
2. (Plate#, DateOfInspection, InspectionResult)
3. (owner-ssn, owner-address, owner-phone)