Pset 3 - Due: Wednesday, September 25

For each of the below, part b refers to the results of part a, and part c refers to the results of part b. Any changes made during the previous steps should be considered in the steps that follow

1. R(A, B, C, D, E, F, G, H)
Functional Dependencies:

$$A \to D, E$$
$$C \to G$$
$$A, C \to H, F.$$

(a) Is this relation in 1NF? If not, explain why not, then make the necessary changes to fix it.

(b) Is this relation in 2NF? If not, explain why not, then make the necessary changes to fix it.

(c) Is this relation in 3NF? If not, explain why not, then make the necessary changes to fix it.

Remark (normal forms)**.** A relation is in 1NF iff

1. All values are atomic

This signals that the current primary key is sufficient in uniquely identifying each tuple.

A relation is in 2NF iff

1. It is already in 1NF
2. No non-prime members are functionally dependent on a subset of the primary key.

A relation is in 3NF iff

1. It is already in 2NF
2. No transitive dependencies in the same relation

Remark (Patterns to achive normality)**.** If a relation is not in 1NF, we must identify a new primary key that uniquely identifies each tuple and allows values to be atomic.

If a relation is not in 2NF, we follow a decomposition pattern.

1. Start with the original relation, and the FD that causes the violation.
2. The attributes on the RHS of the FD are removed from the original relation and placed into a newly created relation that has the FD's LHS as its primary key. A foreign key links the attribute from the LHS in the original table (the LHS is not removed) to the corresponding tuple in the new table, where it is the primary key.

If a relation is not in 3NF, we follow the same decomposition pattern as with 2NF

1. Start with the original relation, and the FD that causes the violation.
2. The attributes on the RHS of the FD are removed from the original relation and placed into a newly created relation that has the FD's LHS as its primary key. A foreign key links the attribute from the LHS in the original table (the LHS is not removed) to the corresponding tuple in the new table, where it is the primary key.

a.) This relation is in 1NF because all the values are atomic, no changes here

b.) To check if this relation is in 2NF, we must first identify the primary key. We see from the functional dependencies, A, B, and C together determine all the attributes in the relation. Thus, the primary key is $\{A, B, C\}$. Since G is functionally dependent on a subset of the primary key $C \rightarrow G$, where $\{C\} \subset \{A, B, C\}$. And $\{D, E\}$ is functional dependent on A, $A \rightarrow D, E$, with $\{A\} \subset \{A, B, C\}$. Lastly, $\{H, F\}$ are functional dependent on A, C, where $\{A, C\} \subset \{A, B, C\}$ This relation is not in second normal form.

To address these issues, we create three new relations, will will now have four relations.

R(A, B, C)
R_α(C, G)
R_β(A, D,E)
R_γ(A, C, H, F).

c.) Since there are no transitive dependencies, these four new relations are all in third normal form

2. Property(id, county, lotNum, lotArea, price, taxRate, (datePaid, amount))
Functional Dependencies:

$$\begin{aligned} \text{id} &\rightarrow \text{county, lotNum, lotArea, price, taxRate} \\ \text{lotArea} &\rightarrow \text{price} \\ \text{county} &\rightarrow \text{taxRate} \\ \text{id, datePaid} &\rightarrow \text{amount.} \end{aligned}$$

a.) This relation is not in 1NF because (datePaid, amount) signifies non-atomic values. Thus, we need to determine a primary key to address this. Looking at the functional dependencies we see id and datePaid together deterimine all the other attributes, so we select {id, datePaid} as the new primary key. The relation is then

Property(<u>id</u>, county, lotNum, lotArea, price, taxRate, <u>datePaid</u>, amount)

This relation is now in 1NF

b.) This relation is not in 2NF, we see that county, lotNum, lotArea, price, and taxRate all depend on a subset of the primary key. Following the decomp pattern described above, we get the new relations

Property(<u>id</u>, <u>datePaid</u>, amount)
Info(<u>id</u>, county, lotNum, lotArea, price, taxRate).

c.) The second relation is not in third normal form because we have transitive dependencies

$$\begin{aligned} \text{lotArea} &\rightarrow \text{Price} \\ \text{county} &\rightarrow \text{taxRate.} \end{aligned}$$

Thus, we create two new relations. We will now have four relations

Property(<u>id</u>, <u>datePaid</u>, amount)
Info(<u>id</u>, county, lotNum, lotArea)
PriceInfo(<u>lotArea</u>, price)
CountyTax(<u>conty</u>, taxRate).

3. Pharmacy(patient_id, patient_name, address, (Rx_num, trademark_name, generic_name, (filldate, num_refills_left), num_refills))

Functional Dependencies:

$$\begin{aligned} \text{patient\_id} &\rightarrow \text{patient\_name, address} \\ \text{patient\_id, Rx\_num} &\rightarrow \text{trademark\_name, generic\_name} \\ \text{Rx\_num} &\rightarrow \text{num\_refills} \\ \text{Rx\_num, filldate} &\rightarrow \text{num\_refills\_left.} \end{aligned}$$

a.) This relation is not in 1Nf, (Rx_num, trademark_name, generic_name, (filldate, num_refills_left), num_refills) signifies non-atomic values. Thus, we must choose a primary key to uniquely identify each tuple and get rid of the non-atomic value. Looking at the FDs, we see that patient_id, Rx_num, and filldate together determine all attributes. Thus, the primary key is {patient_id, Rx_num, filldate}, and the relation is

Pharmacy(patient_id, patient_name, address, Rx_num, trademark_name, generic_name, filldate, num_refills_left, num_refills).

b.) This relation is not in 2NF because there are nonprime attributes functional dependent on a subset of the primary key, to fix this we follow the decomposition pattern

Pharmacy(patient_id, Rx_num, filldate)
PatientInfo(patient_id, patient_name, address)
ScriptInfo(patient_id, Rx_num, trademark_name, generic_name)
RefillAllowed(Rx_num, num_refills)
RefillLeft(Rx_num, filldate, num_refills_left).

c.) These relations are in 3NF because there are no transitive dependencies

4. Company(EmpID, EmpName, EmpAddr, (ProjID, ProjName, MgrID, MgrName, HoursWorked))

Functional Dependencies:

$$\begin{aligned} \text{EmpID} &\rightarrow \text{EmpName, EmpAddr} \\ \text{ProjID} &\rightarrow \text{ProjName, MgrID, MgrName} \\ \text{EmpID, ProjID} &\rightarrow \text{HoursWorked} \\ \text{MgrID} &\rightarrow \text{MgrName.} \end{aligned}$$

a.) This relation is not in 1NF because (ProjID, ProjName, MgrID, MgrName, HoursWorked) signafies non-atomic values. Looking at the FDs, we see that EmpID, and ProjID together get all all eight attributes. Thus, the primary key is {EmpID, ProjId}, and the relation becomes

Company(EmpID, EmpName, EmpAddr, ProjID, ProjName, MgrID, MgrName, HoursWorked).

b.) This relation is not in 2NF because some nonprime attributes are functionally dependent on a subset of the primary key. After decomposition, we get the relations

Company(EmpID, ProjID, HoursWorked)
ProjMan(ProjID, ProjName, MgrID, MgrName)
Emp(EmpID, EmpName, EmpAddr).

c.) The second relation is not in 3NF because there is a transitive dependency MgrID $\rightarrow$ MgrName. Thus, we decompose. The final relations are

Company(EmpID, ProjID, HoursWorked)
ProjMan(ProjID, ProjName,)
Manager(MgrID, MgrName)
Emp(EmpID, EmpName, EmpAddr).

5. StockExchange(Company, Symbol, HQ, Date, ClosePrice)
Functional Dependencies:

$$\text{Symbol, Date} \rightarrow \text{Company, HQ, ClosePrice}$$
$$\text{Symbol} \rightarrow \text{Company, HQ}$$
$$\text{Symbol} \rightarrow \text{HQ}.$$

a.) This relation is already in 1NF

b.) This relation is not in 2NF. First, let's identify the primary key. We see that Symbol and Date together determine all attributes. Thus, {Symbol, Date} is the primary key for this relation.

StockExchange(Company, <u>Symbol</u>, HQ, <u>Date</u>, ClosePrice).

Now that we have the primary key, we see that company and HQ depend on a subset of the primary key, so we decompose.

StockExchange(<u>Symbol</u>, <u>Date</u>, ClosePrice)
SymbolInfo(<u>Symbol</u>,Company)
CompanyLoc(<u>Symbol</u>, HQ).

c.) These relations are already in 3NF, no transitive dependecies.