CS443 - Lab 3

30/30

Question 1:

Suppose that our database has the following table.

Person

Con ID	Con Name	Con Pop	Con Size	State code	State Name	State Rgn	State Size	State Pop	Cty Code	Cty Name	Cty Size	Per SSN	Per Name	Per Age	Per DofB	Per Add

Field Explanation:

Con: Stands for Country
Pop: Stands for population
Rgn: Stands for region (like west, east, central, etc.)
Cty: Stands for City
Per: Stands for Person
DofB: Stands for date of birth
Add: Stands for Address

It is assumed that

- Every country in the world has a different country ID
- Every city in the world has a different city code
- Every state in the world has a different state code, and
- Every person in the world has a different SSN
- Every person in the world has only one citizenship and has only one address

Other Assumptions:

- There is no village, county, area, etc. A country consists of several states and each state has several cities
-

1) Based on the above assumptions, what do you choose to be the primary key of Person table? Why?

Given the assumptions above, the most suitable primary key for the Person table is the SSN number. This is because of the uniqueness of the SSN (does not repeat), ensuring that each table entry is distinct, as every individual is associated with a single address and thus, a singular record in the table.

2) Explain the anomalies exist in the Person table. Choose only one example of insert anomaly, one example of delete anomaly and one example of update anomaly. Note that update does not mean adding or deleting records. It only refers to modifications of values in some rows of the table.

Insert Anomaly: one example would be if we attempted to add a new city to the table, if we did that, the primary key would be null, and therefore an insert anomaly would occur.

Delete Anomaly: Just like adding a city creates an anomaly, removing a city from the table would not only delete the city itself but also erase all data regarding individuals who resided in that city, leading to a loss of information.

Update Anomaly: if the population changes within the table, we would have to update more than one row.

3) Normalize the table; create as many as tables necessary such that all new tables are in third normal form. All the transitive and derived dependencies must be removed.

Transitive:

Population: (ConPop and StatePop)
If we assume that everyone in the world is within this database, we can derive both population columns in the Person table, Country and State. We can do this because, for both, we can just sum up the number of people that we would like to get. We can get the population of a city, by adding up everyone who lives in that city. And from there, it is the same. Summing up cities in a state will give the state population. And summing up all the states in a country will give the country's population.

Size: (ConSize, StateSize)
When calculating the sizes, we can do a similar tactic as with population, however, we cannot get rid of city size because we would have no way to calculate that data. Therefore to get state size, we sum up city sizes. And for country size, we sum up state size.

Person Age:
And finally, we are able to calculate a person's age from their date of birth, therefore having a column for age is not needed.

Tables after accounting for both transitive and derived dependencies:

Person (PerSSN, PerName, PerDofB, PerAdd, CtyCode*)
City (CtyCode, CtyName, CtySize, StateCode*)
State (StateCode, StateName, StateRgn, ConID*)
Country (ConID, ConName)

4) Draw your ERD based on fully normalized table (Reverse Engineering).