Problem 1.

Consider a relation \( R(A, B, C, D) \) with FD’s \( AB \rightarrow C \), \( C \rightarrow D \), and \( D \rightarrow A \).

(a) Show that \( \{A, B\} \) is a key of \( R \) (remember a key has to be minimal).

(b) What are the other keys of \( R \)? (Hint: \( B \) must be in every key of \( R \); why?)

(c) \( D \rightarrow A \) is a BNF violation. Using this violation, we decompose \( R \) into \( R_1(A, D) \) and \( R_2(B, C, D) \). What are the keys of \( R_1 \)?

(d) What are the FD’s that hold in \( R_1 \)? Do not list them all; instead, give a set of FD’s from which all other FD’s in \( R_1 \) follow. This set of FD’s is called a basis. When checking for BCNF violations, it suffices to check just the basis.

(e) Is \( R_1 \) in BCNF? Briefly explain why.

(f) What are the keys of \( R_2 \)? (Hint: There is more than one.)

(g) What are the FD’s that hold in \( R_2 \)? Again, do not list them all; instead, give a basis.

(h) Is \( R_2 \) in BCNF? If yes, briefly explain why. Otherwise, decompose further until all decomposed relations are in BCNF, and then show your final results.

Problem 2.

Consider again the beer drinker’s database from Homework #1 (slightly augmented):

\[
\begin{align*}
\text{Drinker} & \left( \text{name, address} \right), \\
\text{Bar} & \left( \text{name, address} \right), \\
\text{Beer} & \left( \text{name, brewer} \right), \\
\text{Frequents} & \left( \text{drinker, bar, times\_a\_week} \right), \\
\text{Likes} & \left( \text{drinker, beer} \right), \\
\text{Serves} & \left( \text{bar, beer, price} \right).
\end{align*}
\]

Run `~cps116/examples/db-beers/setup.sh` to setup a database with some sample data.

For the SQL database schema, please refer to the file `create.sql` in the same directory.

Write SQL statements to answer the following queries. Make sure that result rows are ordered and contain no duplicates. Use \textit{DISTINCT} only when necessary.

Write all your queries in a file named `hw2-2.sql`. When you are done, run “\texttt{db2 -tf hw2-2.sql > hw2-2.out}” (you may need to run “\texttt{db2 connect to cps116}” before that and “\texttt{db2 disconnect all}” afterwards). Then, print out files `hw2-2.sql` and `hw2-2.out` and turn them in together with the rest of the assignment.

(a) Find all drinkers who frequent James Joyce Pub.

(b) Find all bars that serve both Amstel and Corona.

(c) Find all bars that serve at least one of the beers Amy likes for no more than $2.50.

(d) For each bar, find all beers served at this bar that are liked by none of the drinkers who frequent that bar.

(e) Find all drinkers who frequent only those bars that serve some beers they like.

(f) Find all drinkers who frequent every bar that serves some beers they like.

(g) Find those drinkers who enjoy exactly the same set of beers as Amy.
(h) For each beer, find the bars that serve it at the lowest price.
(i) For each beer, find its average price and popularity (measured by the number of drinkers who like it). Sort the output by average price.
(j) Every time when Dan goes to a bar, he buys the most expensive beer he likes that is served at this bar. If there is more than one such beer, he buys just one of them. If the bar does not serve any beer he likes, he will go for a glass of wine. Find the amount of money Dan spends every week buying beers in bars.

Problem 3.

For this problem, you may need to refer to Problem 1 of Homework #1 (available on course Web site) and the sample solution (available only in hardcopies). Recall that the sample solution consists of the following tables (we ignore Reviews for this problem):

Automobile (VIN, model, make, year, color, mileage, body_style, sellerID)
Dealer (sellerID, name, address, phone)
IndividualSeller (sellerID, phone, email)

Keep all SQL statements you write for this problem in a file named hw2-3.sql. You can use “@” instead of “;” as the statement termination character in this case because of the triggers you are going to write in (b). When you are done, run "db2 –td@ -f hw2-3.sql > hw2-3.out". Then, print out files hw2-3.sql and hw2-3.out and turn them in together with the rest of the assignment.

(a) Create the schema according to this design using CREATE TABLE statements. Choose appropriate data types for your columns, and remember to declare any keys, foreign keys, NOT NULL, and CHECK constraints when appropriate.

(b) Note that any Automobile.sellerID must be a Dealer.sellerID or IndividualSeller.sellerID. Also, a Dealer.sellerID cannot be an IndividualSeller.sellerID, and vice versa. It is not possible to declare these constraints as straightforward key and foreign key constraints. Instead, write triggers to enforce the following:

- An update or insertion of an Automobile row is rejected if the new sellerID is found in neither Dealer nor IndividualSeller.
- An update or deletion of a Dealer or IndividualSeller row is rejected if the old sellerID is found in Automobile.
- An update or insertion of a Dealer (or IndividualSeller) row is rejected if the new sellerID is found in IndividualSeller (or Dealer, respectively).

(c) Start with empty tables. Write INSERT, UPDATE, and DELETE statements to illustrate that the triggers you wrote for (b) are working properly. More specifically:

- The first statement should attempt to insert a row into Automobile but should be rejected by your triggers.
- The second statement should insert a row into Dealer successfully.
- The third statement should attempt to insert a row into IndividualSeller but should be rejected by your triggers.
- The fourth statement should insert a row into IndividualSeller successfully.
• The fifth statement should insert a row into Automobile (with sellerID referring to a Dealer) successfully.
• The sixth statement should update the Automobile row’s sellerID to refer to an IndividualSeller successfully.
• The seventh statement should attempt to update the Automobile row’s sellerID but should be rejected.
• The eighth statement should attempt to delete the IndividualSeller but should be rejected.
• The ninth statement should delete the Automobile row successfully.
• The tenth statement should delete the IndividualSeller row successfully.
• The eleventh statement should delete the Dealer row successfully.