Homework 1
Question 2 solutions by teaching staff of ComS 363

a.

Answer: No.
Reason: The inserted record has no ingredients associated but an iid value is required since it is part of the primary key. A primary key must not have a null value in any of its attributes.

b.

We have looked at how to store data in multiple relations where we can join relations and get the same result but with less redundancies. This exercise tests your understanding of the primary key and foreign key. We have practiced this concept in Week 3 when looking at rdbdesign_questions.sql.

Design 1: The information about which food has which ingredient is kept in ingredient table with iid as the primary key.

Answer: No.
Example/Reason: For example, if the 1st row of Figure 2 is already stored in the ingredient table, the 3rd, the 7th rows, and the second to last row of Figure 2 cannot be stored since they have the same iid value of 23 as the first row does.

Longer explanation: The rows in Figure 2 with the same iid value cannot be stored in the same instance of the ingredient table in Design 1. Design 1 allows an ingredient to belong to at most one food since iid is the only attribute of the primary key of the ingredient table; fid is an attribute that can store at most one value.

Design 2: The information about which food has which ingredient is kept in the recipe table; however, fid is the only attribute of the primary key.

Answer: No
Example/Reason: For example, rows 3rd, 4th, and 5th in Figure 2 cannot all be stored in the same instance of the recipe table in Design 2 since these rows have the same fid value of 27. Only one of them can be stored in a database instance since fid is the primary key of the recipe table and iid is an attribute that can store at most one value. Due to the primary key constraint, there cannot be more than one row with the same fid value.

Design 3: The information about which food has which ingredient is kept in the food table where fid is the primary key.

Answer: No.
Example/Reason: For example, rows 3rd, 4th, and 5th in Figure 2 cannot all be stored in the same instance of the food table in Design 3 since they share the same fid value of 27. Only one of these rows can be stored in a database instance since fid is the only attribute of the primary key of the food table.

The rows in Figure 2 with the same fid value but different iid values cannot be stored in the same instance in the food table of Design 3.

Design 4: The information about which food has which ingredient is stored in the recipe table with both fid and iid as the primary key. Each row in this table has a unique combination of fid and iid values. This design allows more than one row with the same fid value as long as the iid values of these rows are different. It also allows more than one row with the same iid value as long as the fid values are different.

Answer: Yes

Reason: In the recipe table, fid and iid attributes form the primary key. Therefore, a food item is allowed to have multiple ingredients and an ingredient is allowed in multiple food items with the associated amount as in the food_ingredient table. Joining the three tables based on the primary key of the referenced table(s) and the foreign key of the referencing table(s) together gives the same information as in the food_ingredient table. No attributes in the food_ingredient table are missing.

Additional information not required by students to answer
The foreign key recipe.fid to food.fid makes sure that the recipe.fid value must be a valid fid value in the food table. The recipe.iid is a foreign key to the ingredient.iid, which makes sure that it is a valid iid value in the ingredient table. All functional dependencies are preserved.

The functional dependency {fid,iid}-->{amount} is satisfied in the recipe table. {fid}->{fname} is preserved in the food table and {iid} -> {iname, caloriepergram, category} is preserved in the ingredient table.