CMPSC 312
Database Systems
Fall 2020

Lab 4 Assignment: Restaurant Reservation System

Submit deliverables through your assignment GitHib repository.
Place builder code in `src/` directory

Objectives

To demonstrate proficiency in applying primary and foreign keys for a database system that would be used by a restaurant to make reservations for their famous guests. In this database system you will implement primary and foreign keys as constraints.

GitHub Starter Link

`https://classroom.github.com/a/1GeDyEpz`

To use this link, please follow the steps below.

- Click on the link and accept the assignment
- Once the importing task has completed, click on the created assignment link which will take you to your newly created GitHub repository for this lab,
- Clone this repository (bearing your name) and work locally
- As you are working on your lab, you are to commit and push regularly. The commands are the following.
  - `git add -A`
  - `git commit -m ''Your notes about commit here''`
  - `git push`

Introduction

During our study of primary and foreign keys, we have discussed how primary keys are used to establish non-redundancy in tables where they are implemented. In our discussion of foreign keys, we have discussed how these keys are used to maintain strict relationships between different tables. In this lab, we we further this discussion by demonstrating how these keys can be used to establish non-redundancy and strict relationships for a database which is built to store reservation information for restaurants.

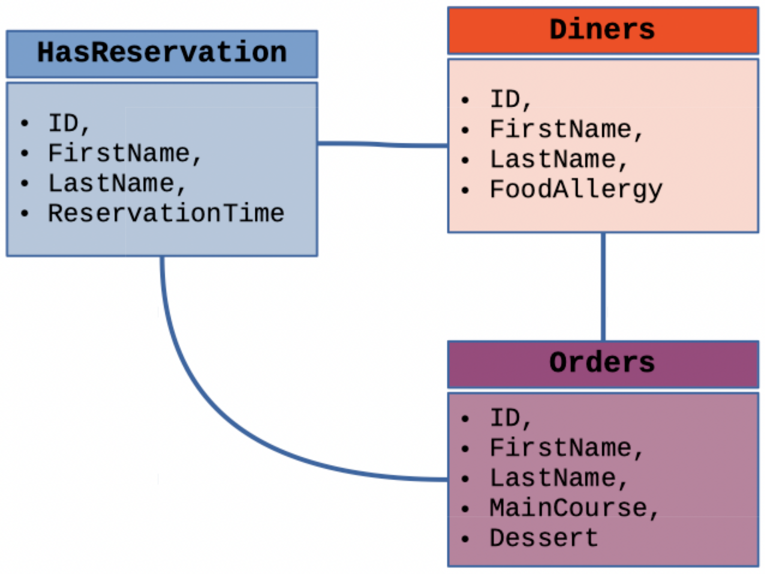

Figure 1: The simplified diagram of the *RestaurantDB* database. The primary and foreign key information has been left out of this diagram. Part of your task is to design these keys.

RestaurantDB

We will be building a database that has been designed for a restaurant which takes reservations on a daily basis for diners who come to eat. The database will have three tables;

1. Table `HasReservation`: Contains an attributes for diners' unique identifiers, the first and last names, and the times of their reservations in the restaurant.
2. Table `Diners`: Contains attributes for diners' unique identifiers, the first and last names, and an attribute (`foodAllergy`) to record any observed food allergies for each diner.
3. Table `Orders`: Contains attributes for diners' unique identifiers, the first and last names, an attribute (`mainCourse`) to record the main course of the diner, and an attribute (`dessert`) to record the type of dessert that the gastronome will enjoy after the main course.

The database is used in the following way; when a diner (the customer) makes a reservation to be seated in the restaurant for dinner, the restaurant's database administrator places the person's name, along with the names of the others in his or her party in the `HasReservation` table. The administrator also records the requested time of the reservation in this table.

When the dining party arrives at the restaurant for their reservation, the host will have knowledge of the unique identifier for each person. The host will populate the Table `Diners`, with the unique identifier of each person, and the first and last names of each member of the party. The host will also ask each person about any food allergies. Food allergy information is to be recorded by the `Diners` Table attribute, `foodAllergy`. **We note, that this table can only be populated when there is already a reservation for the diner.**

The server who takes the order from each of the diners will populate another table `Orders` which stores with the unique identifier of each person, and the first and last names of each member of the party. In addition, the `Orders` table will also contain two attributes to record the `mainCourse` and `dessert`. **This table can only be populated when there is already a reservation for the diner.** The simplified diagram for the restaurant's database can be found in Figure 1.

Building Task

Some code to populate the `Diners` table has been added to your partically completed builder file. Your task is to design and implement this database system for the restaurant. You will have to create all tables and use primary and foreign keys where appropriate. We note that the primary keys of each table are to be used to ensure that there is no redundancy in the database, and the foreign keys are to be used to ensure the relationship that has been described above. Your deliverable will be a builder file that will be used to build the database, and will also populate its tables.

Query Task

The other part of your task is to demonstrate the functionality of your keys. For this, you will write three or four queries of your choice to demonstrate how the primary keys function to unite your tables.

Summary of the Required Deliverables

Please submit your work by pushing it to your GitHub Classroom repository.

1. **Builder file**: `src/builder_restaurantDB.txt` This file will build your database's tables and add data.
2. **Queries file**: `src/myQueries.md` This file contain three or four queries to demonstrate the functionality of your database. Please be creative in your thinking so as to emulate an actual query that a restaurant may use when working with your database system. Please use Markdown when writing this file.

In adherence to the Honor Code, students should complete this assignment on an individual basis. While it is appropriate for students in this class to have high-level conversations about the assignment, it is necessary to distinguish carefully between the student who discusses the principles underlying a problem with others and the student who produces assignments that are identical to, or merely variations on, someone else's work. Deliverables that are nearly identical to the work of others will be taken as evidence of violating Allegheny College's Honor Code.