CSc 460 — Database Design
Fall 2022 (McCann)

http://u.arizona.edu/~mccann/classes/460

Program #4: Database Design and Implementation

Due Dates:

Team Members:	November 21st, 2022, at the beginning of class
Draft E–R Diagram:	November 28th, 2022, at the beginning of class
Final Product:	December 5th, 2022, at the beginning of class

Designed by *Priya Kaushik and Aayush Pinto*

Overview: In this assignment, you will build a database–driven information management system from ground up. We will give you an application domain to work on, your goal is to design the underlying database and define the application functionalities you will provide with the database, and implement this application using Oracle within a text–based JDBC program.

Assignment: In this assignment you are to implement a two–tier client–server architecture.

1. **Database Back-End**, which runs the Oracle DBMS on `aloe.cs.arizona.edu`. Your job is to design the database relational schema, create tables and populate your tables with some initial data. We are requiring that you create an E–R diagram, analyze the FDs of each table and apply table normalization techniques to your schema to justify that your schema satisfies 3NF, and, if possible, BCNF.
2. **JDBC Front–End**, which is the client's user interface. You need to design a text–based application that appropriately handles all the required functionalities. Your client application will run on lectura.

Application Domain: The problem description for the project is as follows:

> Tucson International Airport (TIA) offers the people of southern Arizona flights on several airlines. Your task is to design a database and an associated manipulation / querying application for TIA.
>
> For this assignment, assume that passengers choose from four major Airlines: Delta, Southwest, United, and Alaska. Passengers can get different benefits depending on their category. There are three benefits: Discounts on their meals/beverages on flight if they are a frequent flyer, an extra baggage allowance if they're a student, and a category based on your choice. Everyone other than students are allowed no more than two bags.
>
> Every flight is associated with a unique flight ID, staff (pilots, cabin crew, and ground staff), boarding gate, boarding time, departing time and duration of flight, and route of the flight (i.e., origin for arriving flights and destination for the departing flights). The duration of a flight is anywhere between 1 - 5 hours and flights start and end on the same date. Duration of flight should be in hours and minutes. The data is to be populated for the year 2021. For specifying time, use the 24–hour format (i.e., 7PM = 1900 hours). (Use the Oracle DATE type for date and time storage; information about it is readily available online.) Do not include the arrival times in the flight schedule.
>
> Finally, the passengers' flying history needs to be documented. Some may not have any. Some may have a history with only a few selected airlines or just one.

(Continued...)

This description does not describe every detail. These are the essentials; we expect that your team will create logical and conceptual designs that incorporate all of these features, at minimum. You are free to add additional details that you feel are appropriate.

For each table you create, you need to populate a reasonable number of tuples in order to test your queries adequately. Some data basics are provided in the application domain description; the rest are left for you to determine, based on the scenario and your application's needs. (What is 'reasonable' is difficult to define; a few dozen tuples per relation certainly would be; just a handful per relation may not provide sufficient variety.)

We realize that you are not an expert in this domain, but you have dealt with similar organizations in your life. Hopefully, your team has a wide–enough variety of life experiences that this problem description makes sense. If you have questions, please ask, and the TAs will help you clear things up.

Required functionalities: Within the framework provided above, your system is expected to perform examples of the following operations:

1. *Record insertion*: Your application should support inserting a new data record via a JDBC interface.
2. *Record deletion*: Your application should support deleting an existing data record via a JDBC interface.
3. *Record update*: Your application should support updating an existing data record via a JDBC interface.
4. *Queries*: Your application should support querying your database via a JDBC interface for the problem description given above. You are required to implement the four provided queries as well as at least one query of your own design. Details are provided below.

Specifically, the JDBC application's interface should enable users to:

1. Add, update or delete a passenger and their details, including the passenger's history. When deleting a passenger and their details, the entire row regarding the passenger needs to be deleted. While adjusting a passenger's flights and flight history, make sure the passenger is not on flights with overlapping times.
2. Insert, update or delete flights, staff and their details. When updating information about flights, the user is allowed to update everything except the flight ID. When changing flight times, if a time–overlap is created, do not let the change take place. When deleting a flight and its details, the entire row regarding the flight needs to be deleted.

Here are the queries that your application is to be able to answer:

1. Display the list of distinct passenger names, who took flights from all four airlines in the year 2021.
2. For any airline entered by the user, print the list of passengers, with the number of checked–in bags. Sort the list in ascending order of the number of checked–in bags. Display the output grouped by flights for a particular date in March (input by the user).
3. Print the schedule of flights for a given date in June (input by the user). The schedule should contain the flight number, gate number, name of the airline of that flight, boarding time, departing time, duration of flight, route of the flight (i.e. origin for arriving flights and destination for the departing flights). Sort the schedule in ascending order of the boarding time.
4. Print the list for the three categories of passengers (Student, Frequent Flyer, Category of your choice) for each of the following queries for a particular airline (of your choice) who:
   - Traveled only once in the month of March
   - Traveled with exactly one checked in bag anytime in the months of June and July
   - Ordered snacks/beverages on at least on one flight

   (There will be three result tables per category, nine tables total.)

(Continued...)

Working in Groups: In industry, such a project is usually the work of multiple developers, because it involves several different components. Good communication is a vital key to the success of the project. This assignment provides an opportunity for just this sort of teamwork. Therefore, we are accepting team sizes of between two and four members (inclusive). **Working alone is not permitted.**

Early on, you will need to agree on a reasonable workload distribution plan for your team, with well–defined responsibilities, deliverables, and expected completion dates. Such a plan will minimize conflicts and debugging effort in the actual implementation.

Late days: Late days can be used on this assignment, but only on the third due date. How many a team has to use is determined as follows: Team members total their remaining late days, and divide by the number of members in the team (integer division), producing the number of late days the team has available, **to a max of two days late**. (Why a max of two? The TAs need to get grading done soon after the due date, you need time to study for your final exams, and the department has a rule about assignments needing to be due before the start of finals.) For example, a team whose three members have 1, 1, and 3 late days remaining have $\lfloor \frac{1+1+3}{3} \rfloor = 1$ late day to use, if needed.

Hand In: Here are the 'deliverables' for each of the assignment's three due dates:

1. *Team Composition*: By the first due date (see the top of the front page of this handout), one member of your team must create a PRIVATE post on Piazza using the "program4" folder with the names and NetIDs of the members of your team. Failure to do so by the start of class on this date will cost your team the corresponding points listed in the Grading Criteria section (below).

2. *E–R Diagram*: As stated in the Assignment section, your team will need to create an E–R diagram that describes your database design. Before the second due date, your team will need to prepare a draft of your E–R diagram **and** a member of your team will need to submit it through `turnin` to the `cs460p4` folder. The purpose of this requirement is to allow the TAs to review your schema and make suggestions for improvement. The sooner you create your design and discuss it with the TAs, the more time you will have to refine your final E–R diagram. If TAs need further explanation of your E–R Diagram, they'll send out an email to make an appointment to have an additional meeting.

3. *Final Product*: On or before the third due date, a member of your team must submit a `.tar` file of your well–documented application program file(s) via turnin to the folder `cs460p4`. The tar file should contain all of the following:
   (a) The source code for your application.
   (b) A PDF file called "design.pdf" containing the following sections in this order:
      i. *Conceptual database design*: Your final E–R diagram along with your design rationale and any necessary high–level text description of the data model (e.g., constraints, or anything you were not able to show in the E–R diagram but that is necessary to help people understand your database design).
      ii. *Logical database design*: The conversion of your E–R schema into a relational database schema. Provide the schemas of the tables resulting from this step.
      iii. *Normalization analysis*: For each of your entity sets (tables), provide all of the FDs of the table and justify why your the table adheres to 3NF / BCNF.
   (c) A `ReadMe.txt` describing:
      i. Compilation and execution instructions, to enable the TAs to execute your application and exercise the required functionalities.
      ii. The workload distribution among team members (that is, which people were responsible for which parts of the project).

   In addition, each team must schedule a time slot (∼20 minutes) to meet with a TA, demonstrate your system, and answer questions about it. Closer to the final due date, we'll let you know how to sign up.

(Continued...)

Grading Criteria: Total: 100 points

1. Team Composition (1st due date): 5
2. Complete E–R Diagram Draft (2nd due date): 20
3. Final Submission (3rd due date): 75
   (a) Coding / Implementation: 55
      - Documentation 15
      - Style and organization 10
      - Record insertion: 5
      - Record deletion: 5
      - Record update: 10
      - Record query: 10
   (b) Database design: 20
      - Final E–R diagram: 10
      - Normalization analysis: 10

Grading Notes:

1. Unless we receive verifiable information about inadequate contributions, each member of a team will receive the same score on this assignment.
2. We won't put much weight at all on the appearance of the text application; concern yourselves with the application's functionality instead. The main details of the assignment are the DB design and its implementation.

Final Detail

- Priya and Aayush are the clients for this design and implementation project. As such, you should direct questions about the assignment details and expectations to them, not to Prof. McCann.