# ISTE-330 Database Connectivity and Access

# Spring 2024

# Project Description

**(Deadline: will be announced)**

# Mission

This is a comprehensive exercise where you will demonstrate your knowledge and understanding of the database connectivity concepts covered throughout this course.

# Description

Define a business scenario (or other context) where a relational database is needed. The business could be your friend’s startup, an established business (whether physical or virtual, i.e., website), or a realistic made-up scenario. It is possible that a flawed database solution exists, but you want to fix the current solution by proposing an efficient and secure database.

Define a minimum of four core business requirements that the system users should be able to accomplish. Examples include: enrolling new students, registering new items, generating reports of transactions in the last month, creating new tables (by database admins only), defining user roles (by database admin only), and transferring credits/points/funds across accounts. You must demonstrate to me how your system accomplishes those core requirements.

The following layers and features must be included in your system:

1. (20 points) A presentation layer Java code that operates entirely through an interface (do not use standard I/O to enter/output data). The interface has the following features:
   1. Buttons to connect and disconnect to and from the database.
   2. Text fields to input a user’s ID and password. Feedback must be provided to the user indicating a successful or failed login attempt.
   3. Users are of two types: admin or client. You must verify the user type when they successfully login. Based on the user type, you will design your interface to allow the user to do their permitted tasks.
      1. An admin can:
         1. See any table, but cannot make any modifications.
         2. Add a new client.
         3. Add a new admin.
      2. A client can:
         1. See their own information from one or multiple tables.
         2. Make a transfer of credits/points/money by specifying the source account, amount, destination user, destination account.
2. (10 points) A business layer Java code that defines a method for each core business requirement.
3. (40 points) A data layer that implements the various business tasks, such as connectDB(), disconnectDB(), login(), transferMoney(), showAccInfo(), printReport(), unsubscribe(), modifyAccount(), etc.
4. (10 points) An SQL script that creates your database and all its tables and their relationships, and populates those tables.
5. (20 points) An SQL script with at least two stored procedures: one to display a table when given a table name, and another one to perform a transfer of credits. Any stored procedure will be called from the data layer. Your business scenario must have a “transfer” feature.

**Important notes about login passwords and money transfer**:

Figure 1: The bankdb database

* **Passwords**: A user’s password is never stored in plaintext. Instead, the hash value of the password is stored in the database. Use SHA-1 to either store a new user’s password or verify an existing user’s password, as follows:

User provides their plaintext password 🡪 calculate its hash value 🡪 check if equal to stored hash value.

* **Money transfer**: This is a multi-step process that requires you to use transactions, as follows:

Start a transaction 🡪 deduct the amount from the source account 🡪 add the credits/money to the destination account 🡪 check if the balance in the source account is negative 🡪 rollback or commit 🡪 end transaction.

Remember, your code must catch exceptions and must account for semantic or logical errors, such as showing a proper error message if the destination account does not exist.

# Submitting Your Work

This is a group-based assignment. Form a group of three students, then join an available group under myCourses 🡪 Groups. All team members must present their work to me by the deadline indicated on the cover page of this document; failure to do so will result in a **zero** on the whole project for those who do not attend. Your presentation will be in the form of a demo of your program.

Due to time constraint, those who do not present by the deadline, may present during the class after, but must submit everything by the deadline on the cover page of this document.

Your group number, name, and date must show in the following places:

1. At the top of each Java and SQL file as a comment in line 1.
2. In the first line on the Java interface.
3. After establishing connection to the database in the standard output.

Please submit your work to myCourses 🡪 Assignments 🡪 Project. The following must be submitted:

1. At least three error-free Java files: presentation, business, and data layer files. Name the files <Group#PL.Java>, <Group#BL.Java>, <Group#DL.Java>, respectively. Replace “#” with your group number.
2. One SQL script that creates and populates a database. Name the file <Group#DB.SQL>.
3. One SQL script containing stored procedures. Name the file <Group#SP.SQL>.
4. A simple report, as follows:
   1. Cover page with assignment detail, your group, name, deadline, course, etc.
   2. Screenshots of all the outputs, steps, and results explained clearly.
   3. Name this file <Group#Report.docx>

Turnitin will be active, and you will be able to see your Similarity Score. High Similarity Scores (i.e., above 25%) will result in an automatic mark of zero. You have the option to resubmit your report if you wish to modify it before the deadline. The portion of the Similarity Score that generates from copying parts of this report or listing references, does not contribute toward the 25% limit. Moreover, you are not allowed to use AI tools to draw or write any component of your report. Your maximum AI score limit is 5%. This percentage does not mean 5% of your report is AI-generated; rather, the percentage is to account for false positives.

# Mark Distribution

The weights of all the tasks are summarized in the below table. The details of each task are provided to you under section Description.

|  |  |
| --- | --- |
| Task | Weight |
| (1) Java presentation layer | 20% |
| (2) Java business layer | 10% |
| (3) Java data layer | 40% |
| (4) SQL database | 10% |
| (5) SQL stored procedures | 20% |
| **Total** | **100%** |

# Rubric

## Your work in each task will be evaluated against the below rubric.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Rubric | R1 | R2 | R3 | R4 | R5 |
|  | * Code does not run and contains many major errors, or * screenshots and explanations are not provided. | * Code does not run but contains evidence of attempting the required functionalities, and * screenshots and explanations are all provided. | * Code is error-free and only attempts some of the required functionalities, and * screenshots and explanations are all provided. | * Code is error-free and attempts all of the required functionalities but does not account for faulty scenarios, and * screenshots and explanations are all provided. | * Code is error-free and attempts all of the required functionalities and catches all faulty scenarios, and * screenshots and explanations are all provided. |
| **Points** | **0** | **40** | **60** | **80** | **100** |