CSE 305 -- Principles of Database Systems

Fall 2022

Requirements Specification for the Database Programming Project

Introduction

In this project, you will design and implement a relational database system to support the operations of an **on-line stock trading system**, along the lines of E*TRADE, TD Ameritrade, and many others. You will use HTML for the user interface, for the database server, and Java, Javascript, and JDBC for connectivity between the user interface and database server. The database server is accessible from the PCs in both the undergraduate and graduate Transaction Processing labs (Old CS Building, Rooms 2114 and 2126), on which you will be given a MySQL account. You can also simply install MySQL on your laptop (highly recommended).

Please see the Transaction Processing Lab web site for general information about the Transaction lab and its policies, and the lab's MySQL web site for information on how to access MySQL in the Transaction lab and account information.

If you own a laptop, you are encouraged to develop as much of the code as possible (if not the entire project) on your PC, to ease the congestion in the TP lab. For these purposes, I suggest you use MySQL Server or SQL Server.

For MySQL, you can download a copy of MySQL server from the MySQL web site. MySQL is a good RDBMS for learning SQL, has very low resource requirements, is open-source, and has JDBC drivers. Also, it is SQL-99 compliant so it will be more in tune with ANSI SQL standards as opposed to SQL Server, which has some proprietary semantics embedded over the standard query language.

SQL server is a relational database server produced by Microsoft, based on the ANSI SQL-92 standard. You can get a **free** copy of the SQL Server Developer Edition, for 2005 and 2008, from the Stony Brook DreamSpark, formerly known as the Microsoft Developer Network Academic Alliance (MSDNAA), web site.

You are to work in **teams of three**. Many of you will form teams of three on your own. If not, do not worry. Please attend class on a regular basis, and I will make sure that every student is placed in a team of three.

Getting Started

Here are two links to help you get started on the course project.

1. Very Basic Introduction to Stock Trading
2. E*TRADE Conditional Orders Tutorial

The second link is to an online tutorial on **Conditional Orders**, a concept I discuss further below.

Project Specification

The basic idea behind your on-line trading system is that it will allow customers to use the web to browse/search the contents of your database (at least that part you want the customer to see) and to trade stocks over the web. In this regard, it is a lot like the on-line trading system E*TRADE. So visit this site to get ideas as to what your system should look like.

Your database system must be based on the specifications and requirements that follow.

1 System Users

The users of your system will be the customers that use your system to trade stocks and pay fees for doing so, customer representatives who provide customer-related services, and the site's manager. You should assume that the computer knowledge of the users is limited (say, that of a typical AOL subscriber), and thus your system must be easy to access and operate. Customers of a stock-trading system are sometimes also known as *clients*, so I shall use these two terms interchangeably.

2 Required Data

The data items required for the stock-trading database can be classified into four categories: *orders*, *stocks*, *customers* and *employees*, where an *order* is an order to buy or sell a certain number of shares of a particular stock at a certain price.

This classification does not imply any particular table arrangement. You are responsible for arranging the data items into tables, determining the relationships among tables and identifying the key attributes. In addition, you should include indices in your tables to speed up query processing. You should base your choice of indices on the type and expected frequency of the queries outlined in Section 3. Finally, you should specify and enforce integrity constraints on the data, including referential integrity constraints.

As I mentioned in class, you will first create an E-R diagram of your online trading system before developing your relational model. Details of this assignment will soon become available on blackboard.

2.1 Order Data

This category of data should include the following items:

1. Order ID
2. Stock Symbol(GM, GE, IBM, etc.)
3. Order Type (Buy, Sell)
4. Number of Shares
5. Customer Account Number (of the buyer or seller)
6. Date/Time (the order was placed)
7. Transaction Fee
8. Price Type (Market, Market on Close, Trailing Stop, Hidden Stop)
9. Employee ID

An order is the mechanism a customer uses to buy or sell a certain number of shares of a particular stock at a certain price. A transaction fee of 5% is associated with every order.

Your online trading system will also support Conditional Orders such as a *Trailing Stop* or *Hidden Stop* order. A Trailing-Stop order uses a trailing stop to automatically decide when to place a sell order. A trailing stop is set at a percentage or dollar amount below the stock's current market price, and a sell order will be placed if and when the share price falls to the trailing stop. The trailing stop is adjusted as the share price fluctuates.

In contrast, a Hidden-Stop order uses a fixed hidden-stop price to automatically decide when to place a sell order. That is, a sell order will be placed if and when the stock price falls to the hidden stop.

To understand conditional orders better, suppose that you have just bought 1000 shares of GM at $50.00, and you decide that you only want to risk $5.00 per share on this transaction. Accordingly, you immediately place a hidden-stop order at $45.00. This means that if the price of GM should drop to $45.00, your broker will sell your 1000 shares at a market price of $45.00. The use of a hidden-stop order will therefore pre-determine the maximum loss a trader will incur.

Instead of placing a hidden-stop order on your GM shares, suppose now that you place a trailing-stop order with a trailing-stop value of $5.00 (or, equivalently, a trailing-stop percentage of 10%). Thus, your shares will be sold if the share price drops to $45.00. But instead of declining, the price of GM increases to $60.00; but so does the trailing stop. So now your shares will be sold if the share price drops to $55.00. Thus the trailing-stop technique allows an investor to set a limit on the maximum possible loss without setting a limit on the maximum possible gain, and without requiring paying attention to the investment on an ongoing basis.

To find out more about how conditional orders work, please consult the E*TRADE Conditional Orders Tutorial.

2.2 Stock Data

This category of data should include the following items:

1. Stock Symbol
2. Stock Name
3. Stock Type
4. Share Price
5. Number of Shares

An order involves the purchase or sale of a certain number of shares of a stock at a certain price. Stocks are of a certain type: GM is an automotive stock, IBM is a computer stock, etc. You can populate your database with any kind of stocks you like. We will provide you with all the stock data you need to demo your system to us at the end of the semester.

2.3 Customer Data

The items required for this category include:

1. Last Name
2. First Name
3. Address
4. City
5. State
6. Zip Code
7. Telephone
8. E-mail Address
9. Account Number
10. Account Creation Date
11. Credit Card Number
12. Stock Portfolio
13. Rating

A given customer may partake in any number of stock transaction, either as a buyer or as a seller. A customer may have one or more accounts from which to trade stocks. Associated with each account is a stock portfolio, indicating which stocks (and number of shares) are held in that account. The customer's rating should reflect how active a trader he or she is.

2.4 Employee Data

This category of data should include the following:

1. Social Security #
2. Last Name
3. First Name
4. Address
5. City
6. State
7. Zip Code
8. Telephone
9. Start Date
10. Hourly Rate

3 User-Level Transactions

A database *transaction* can be viewed as a small program (written in the DML) that either updates or queries the database. Transactions that change the contents of the database must do so in a consistent manner. Moreover, transactions should not interfere with one another when running concurrently.

What follows is a breakdown of the user-level transactions that your database system should support. To make sure transactions maintain the integrity of the database, you must write them using the SQL transaction structuring capabilities (i.e., `begin transaction`, `commit transaction`, etc.).

3.1 Manager-Level Transactions

The manager should be able to:

- Set the share price of a stock (for simulating market fluctuations in a stock's share price)
- **Add**, Edit and **Delete** information for an employee
- Obtain a sales report for a particular month
- Produce a comprehensive listing of all stocks
- Produce a list of orders by stock symbol or by customer name
- Produce a summary listing of revenue generated by a particular stock, stock type, or customer
- Determine which customer representative generated most total revenue
- Determine which customer generated most total revenue
- Produce a list of most actively traded stocks

3.2 Customer-Representative-Level Transactions

Customer Representatives should be thought of as stock brokers and should be able to:

- Record an order
- **Add**, Edit and **Delete** information for a customer
- Produce customer mailing lists
- Produce a list of stock suggestions for a given customer (based on that customer's past orders)

3.3 Customer-Level Transactions

Customers should be thought of as online traders and should be able to easily browse your online trading system on the web and place orders to purchase or sell stocks. In particular, they should be able to place a trailing-stop or hidden-stop conditional order, and place an order to buy or sell stocks at market or close-of-market price. While they will not be permitted to access the database directly, they should be able to retrieve the following information:

- A customer's current stock holdings
- The share price and trailing-stop history for a given conditional order
- The share price and hidden-stop history for a given conditional order
- The share price history of a given stock over a certain period of time (e.g., past six months)
- A history of all current and past orders a customer has placed
- Stocks available of a particular type and most recent order info
- Stocks available with a particular keyword or set of keywords in the stock name, and most recent order info
- Best-Seller list of stocks
- Personalized stock suggestion list

4 User Access Control

Your database system should provide controlled access to the data by distinguishing between the different types of users: manager, customer representatives, and customers.

- Customer Representatives should not be able to perform manager-level transactions; however, they should be able to read employee information, except for the hourly rate.
- Customer Representatives should be able to record the receipt of an order from a customer.
- A customer should not be allowed access to other customers' account information, or to any employee information. Also, trailing and hidden stops must be kept private.

5 Utilities

In addition to the transactions described above, the system should provide facilities for:

- Allowing the manager to add and delete users
- Backing up the database files
- A comprehensive **Help** facility, including a topic-driven pull-down Help menu

6 User Interface

HTML and its successors provide facilities for creating pop-up and pull-down menus, value lists, input/output forms, labels and customized reports. You should make use of all of these capabilities, and in the process come up with a system that caters to users with only limited computer knowledge. The information you provide to customers should look professional and inviting.

7 Documentation

You will be required to supplement your completed database implementation with a design document that contains information concerning your design criteria and decisions. The following is a list of some of the information you should include:

- Entity-Relationship (E-R) Diagram of the complete database scheme
- Lucid description of the relational database scheme for yoru online stock trading system, including a discussion of the reasoning behind your design decisions. Make clear how your design supports efficient query processing.
- A list of all functional dependencies in the relational database scheme
- Description of integrity constraints including referential integrity

You will also be required to submit a *Users Guide* that carefully explains how to use all aspects of the system. It should be understandable by non-computer experts. Be sure that the user interface (screen design, menu structure, etc.) is clearly explained.

8 Grading

You will be given three assignments: 1) produce an E-R and relational model of your system; 2) implement (in SQL) and execute all transactions described in the above project specification; and 3) implement (using Java, Javascript and JDBC) the final interactive system to support your online stock trading system. The due date for the first assignment will be announced shortly; as for the the other two assignments, the due dates for all three assignments will be spaced roughly three to four weeks apart.

All documentation should be on-line. You will also be asked to hand-in hardcopies when assignments are due.

9 Collaboration Plan

As stated above, you will be working on the course project in in teams of three. A rough, three-way division of labor for the first project assignment is as follows:

- Teammate 1 will focus stock and order data.
- Teammate 2 will focus on conditional orders.
- Teammate 3 will focus on customer and employee data.

About this document ...

Scott Smolka
Last Modified: Sat Jun 18 11:20:11 EDT 2022