Business Data Management
Lab Exercise

Instructor: Periklis A. Papakonstantinou (periklis.research@gmail.com)
Teaching assistant: Hafiz Asif

Due date: Wednesday, April 5 at 11.59pm. We will set up Blackboard to receive your assignment reports (check the Blackboard announcements).

The exercise consists of three related parts.

Remark: You are encouraged to start doing the implementation parts (start implementing the basic functionality first) and contact Hafiz as soon as you face issues. **Do this at least 10 days before the deadline**.

Part 1 (30 points)

Devise relational algebra queries for the following schema of a database storing information about daily stock prices and basic transactions made by a trading firm. You should define domains so as to make the schema complete.

- STOCK (ticker, exchange)

  ticker: the stock's ticker symbol; e.g. GOOG, AAPL, GE
  exchange: the exchange where the ticker is listed; e.g. NYSE, NASDAQ

- PRICE (ticker, date, close)

  ticker: the stock's ticker symbol
  date: the date of the price information
  close: the closing price of the stock

- BUYnSELL (buy_or_sell,ticker, date, timestamp, value, num_of_shares)

  buy_or_sell: 'BUY' or 'SELL'
  ticker: the stock's ticker symbol
  date: the date of the price information

timestamp: time of the transaction
value: the price of a single share
num_of_shares: number of shares (bought or sold)

Express the following as relational algebra expressions.

i. Find the tickers and all closing prices of all stocks exchanged in 2017.

ii. Find all tickers (i.e. for all dates) whose closing price is both higher than 'IBM' on '3/20/2017' and *no higher* than 'GOOG' on '3/20/2017'.

iii. Find the tickers of all stocks that closed at the highest price on '3/20/2017'.
(we are asking for "all stocks" since there may be more than one with the same "highest price")

iv. Find the tickers of all stocks in 'NYSE' whose closing price on '3/20/2017' was either strictly below $20 or strictly above $100

v. Find all tickers in 'NYSE' of the stocks whose closing price showed the highest increase between '3/20/2017' and '3/21/2017' in 'NYSE' and whose closing price was (in 'NYSE') strictly above $100 for the entire 2017
(we are asking for "all stocks" since there may be more than one with the same increase.
Recall that Relational Algebra does NOT support MAX, MIN, AVG operations.)

Part 2 (60 points)

i. (40 points) Realize the database schema of Part 1 in MYSQL. Specify appropriate domains, keys, ICs, and all types of constraints. Then, implement as SQL queries all the relational algebra queries you devised in Part 1 (note that SQL queries can be "different/simpler" than the corresponding Relational Algebra ones) .

In addition, to the above queries, also do in SQL the following one:
Find the dates where the total price (i.e. price times num_of_shares) of 'AAPL' the firm (i.e. the trading firm which is using this database) sold was higher than what the firm bought in 'NASDAQ'.

(1) What is the main reason for not asking to do the above query in Relational Algebra? Justify your answer. Use a detailed explanation and back it up with examples.

(2) Report the code that creates the database. For all the queries you wrote you should report the results of the queries on the example DB instance listed at the end of this lab exercise, but this example DB instance should be modified as follows. Whenever the answer to the query is empty (return nothing) you should add your own typical instances to the provide DB instance.

ii. (20 points) Make an HTML with a single textbox and a "submit button" where the user can enter a MYSQL query on the database you made in (i). Then, a CGI in Python (or in any other programming language you prefer) will submit this query to a MYSQL database and it will return an HTML file showing a simple table with the result of the query. The first row of the table should list the names of the attributes in the database and the remaining rows should list the tuples the query returned. Report the results of the HTML outputs on the example DB instance listed at the end of this document.

Remark: No special HTML formatting is necessary. The most basic (4–5 commands) HTML commands suffice to do this part. *You do not loose any marks* for a very primitive looking webpage design. For example, when the rows are returned it is okay to just print it in the form "TICKER=GOOG", instead of presenting the results in a nicely printed HTML table. Basic HTML and CGI handling can be found either in the online resources we provided or in the 7th Chapter of our textbook.

Important remark: *It is not required* to use Python for the CGI. Use any programming language you feel comfortable with. If you choose Python then you can receive TA help in how Python integrates with MySQL and Apache.

Part 3 (10 points)

Research the literature about what is a Socket and how Socket programming is used. Explain in no more than 100 words how network programming and sockets come together. In another 100 words explain what is a "server". Finally, write an essay of at most 300 words (in addition to the previous 200) and give a detailed diagram explaining which network services are related to your "Part 2" (above) That is, explain how many and what kind of servers (and what do they do) are involved in the system, how your program gets involved in the process, and what kinds of messages the system is exchanging, and what do we mean by "message".

Bonus points (5)

Extra marks for realizing a non-primitive HTML user interface.

Example of database entries

In your report you should present your results in the following database instance. You must also add at least 5 new entries that make the queries you wrote meaningful.

(remark: when you are experimenting with your code you should change the database)

STOCK

ticker	exchange
AAPL	NASDAQ
GOOG	NASDAQ
MSFT	NASDAQ
IBM	NYSE

Remark: If you wish you can modify the above table by *adding* some new entries.

PRICE

ticker	date	close
AAPL	3/20/2017	$100
AAPL	3/21/2017	$101.5
AAPL	3/22/2017	$106.5
GOOG	3/20/2017	$100
GOOG	3/21/2017	$130
GOOG	3/22/2017	$110
MSFT	3/20/2017	$184.5
MSFT	3/21/2017	$188.5
MSFT	3/22/2017	$210
IBM	3/20/2017	$72
IBM	3/21/2017	$70
IBM	3/22/2017	$10

Remark: You must modify the above table by *adding* indicative entries such that the answers to the queries of Part 1 and 2 appear to be somewhat typical. Do not add more than 5 new entries. Present all added entries in your report.

BUYnSELL

ticker	buy_or_sell	date	timestamp	price	num_of_shares
IBM	BUY	3/20/2017	11:55:00	$273	1100
IBM	BUY	3/21/2017	10:45:00	$271	2400
IBM	SELL	3/22/2017	12:09:00	$270.5	2500
GOOG	BUY	3/20/2017	12:22:00	$86	2200
GOOG	SELL	3/20/2017	14:00:00	$87	1000
GOOG	SELL	3/21/2017	10:22:00	$87.5	1000
GOOG	BUY	3/21/2017	13:28:00	$87	800
GOOG	SELL	3/22/2017	11:45:00	$86	500
AAPL	BUY	3/20/2017	10:01:00	$99	1000
AAPL	BUY	3/20/2017	11:22:00	$99.5	1000
AAPL	BUY	3/21/2017	14:22:00	$100	1000
AAPL	SELL	3/22/2017	14:42:00	$103	3000
MSFT	BUY	3/20/2017	11:45:00	$186	1500
MSFT	SELL	3/21/2017	10:45:00	$188	1000
MSFT	BUY	3/22/2017	12:03:00	$187	5000

Remark: You must modify the above table by *adding* indicative entries such that the answers to the queries of Part 1 and 2 appear to be somewhat typical. Do not add more than 5 new entries. Present all added entries in your report.