York University
EECS3421, Summer 2020

# Assignment 1

*Due: Sat, May 30, 11:59pm*

## Introduction

For this assignment, you will be writing queries in relational algebra on a database. We are providing the schema for you. Later in the course, you will learn about how to develop your own schema based on knowledge of the domain. Even though developing a schema is really the first step in building a database, it is a more advanced task than querying an existing database; this is why we will be learning about it and practising it later.

## Schema

The schema below represents data for an online retailer, which you can think of as a simplified version of eBay, where customers can post their items for sale, and other customers can bid or buyout those items. It attempts to represent the domain well, but in order to keep the assignment manageable, some things are simplified and other details are not represented. For example, the payment details (i.e., paid by which credit card) are not included in the schema. Remember that all your queries will be written with respect to the schema below. Your goal is to produce answers that are correct with respect to this schema, even if they aren't quite correct in the real world.

### Relations

- Item(IID, name, category, description)
  A tuple in this relation represents an item for sale. *IID* is the item's ID, *name* is the name of the item (e.g., iPad Mini), *category* describes the kind of item it is (book, toy, etc.), and *description* gives further details about the item.

- Customer(CID, email, name, address, phone, registeredDate)
  A tuple in this relation represents a customer, who can both sell and bid on items in the online store. *CID* is the customer's ID, *email* is the email address the customer wishes to be contacted at, *name* is their name, *address* is their address, *phone* is their phone number, and *registeredDate* is the date the customer was registered in the store.

- Listing(LID, IID, CID, postedTime, expiryTime, startingPrice, buyoutPrice)
  A tuple in this relation represents a listing in the store. A listing is created when a customer attempts to sell some items, which can be bid by other customers. *LID* is the listing's ID, *IID* is the ID of the item that is for sell in this listing, *CID* is the ID of the customer who posted this listing, *postedTime* is the time when the listing is created, *expiryTime* is the time when the listing is expired, *startingPrice* is the starting price of this listing, and *buyoutPrice* is the price that you can buy out this listing immediately if you pay this amount of money.

- Bid(BID, CID, LID, time, price)
  A tuple in this relation represents a single bidding. *BID* is this bidding's ID, *CID* is the ID of the customer who placed this bidding, *LID* is the ID of the listing that this bidding is associated with, *time* is the time when this bidding is placed, and *price* is the price that this bidding offers.

- Rating(CID, IID, number, comments)
  A tuple in this relation represents a customer's rating of an item. *CID* is the ID of the customer who gave the rating, *IID* is the ID of the item that they rated, *number* is the actual rating, and *comment* is a review comment they may have given along with the rating.

- Reputation(CID, number)
  A tuple in this relation represents the reputation of a customer (imagine that there is a system that rates and updates the reputation of all customers). *CID* is the ID of the customer, and *number* is the actual reputation assigned to the customer.

**Integrity constraints**

- Listing[CID] $\subseteq$ Customer[CID]
- Listing[IID] $\subseteq$ Item[IID]
- Bid[CID] $\subseteq$ Customer[CID]
- Bid[LID] $\subseteq$ Listing[LID]
- Rating[CID] $\subseteq$ Customer[CID]
- Rating[IID] $\subseteq$ Item[IID]
- Reputation[CID] $\subseteq$ Customer[CID]

## Part 1: Additional Integrity Constraints [20% - 4 marks each]

Below are some additional integrity constraints on our schema. Express each of them using the notation from Section 2.5.1 of your textbook. If a constraint cannot be expressed using this notation, simply write "cannot be expressed".

1. No one can bid his or her own listing.
2. The price of a bid has to be higher than the starting price of the listing, and cannot exceed the buyout price.
3. All bids need to be placed before the expiry time and after the posted time of the listing.
4. No one can bid again with a new price if the current highest bidder of the listing is him/herself.
5. Later bids have to be higher than previous ones on the same listing.

## Part 2: Queries [80% - 8 marks each]

Write the queries below in relational algebra, assuming set semantics; so you should not use any of the extended relational algebra operations from Chapter 5 (for example, do not use the extended projection). There are several variations on relational algebra, and different notations for the operations. You must use the notation defined in Section 2.4 of the textbook or the same notation as we have used in class and the slides: $\Pi, \sigma, \rho, \bowtie, \bowtie_{condition}, \times, \cap, \cup, -$. If a question cannot be expressed, simply write "cannot be expressed".

1. Report the name of all sellers, i.e., customers who have sold something. A listing is sold if at least one customer bids before the listing's expiry time.
2. Report the ID and the name of the seller with the highest reputation. If there are ties, report all of them.
3. Report the second highest bid price of listing 14563.
4. Report the name of all customers who have bid at least three times on one listing.
5. A rating is usually not trustworthy if the customer who gave the rating never actually won the auction for that item. Find all customers who gave untrustworthy ratings.
6. Report the listing with the highest starting price, among all sold listings. If there are ties, report all of them.
7. People like to buy cheap things. Find the lowest buyout price of currently listed items (i.e., the cheapest iPad Mini, cheapest iPhone, etc.). Report the item name and its buyout price. When comparing dates you can use the literal 'TODAY' to represent the current date.
8. Report the most rated (rated by the largest amount of customers) item. If there are ties, report all of them.
9. Report all customers who have bid every item listed.
10. Report those categories of which all items are always sold with buyout prices. You can assume that if a buyout happens, then this is registered in the Bid relation (i.e., the price of the buyout appears in the Bid relation).

A few points to keep in mind:

- Do not make any assumptions about the data that are not enforced by the constraints given in the schema above and **the additional ones** given in Part 1. Your queries should work for any database that satisfies those constraints.
- The questions are not in order according to difficulty.
- Assume that every tuple has a value for every attribute. (For those of you who know some SQL, in other words, there are no null values.)
- Remember that the condition on a select operation may only examine the values of the attributes in one tuple (not whole columns), and that it can use comparison operators (such as $\leq$ and $\neq$) and boolean operators ($\vee$, $\wedge$ and $\neg$).

- You are encouraged to use assignment (:=) to define intermediate results, and it's a good idea to add commentary explaining what you're doing. This way, even if your final answer is not completely correct, you may receive partial marks.
- The order of the columns in the result doesn't matter.
- When asked for a maximum or minimum, if there are ties, report all of them.
- Some relations in our schema have attributes related to time and a few queries / questions involve working with them. You may assume that the values of those attributes are expressed as timestamps, therefore you can use comparison operators and do arithmetic on them, *e.g.*,

  timestamp1 > timestamp2

## Submission Instructions

Your assignment must be typed; handwritten assignments will **not** be marked. You may use any word-processing software you like. Many academics use LaTeX. It produces beautifully typeset text and handles mathematical notation well. Whatever you choose to use, you need to produce a final document in *pdf*.

**Important Notes:**

- On the first page of your *pdf* you should declare your team (whether it is a team of one or two students) including *login*, *first name*, *last name*, *email* and *student number*
- If you are working in a team, only one of you should submit.
- You should submit your document *electronically* following the instructions below.
- I highly recommend you use overleaf (www.overleaf.com) for preparing your assignment. It makes it easier to work with LaTeX.

### Electronic Submission

You should submit your work electronically using the *submit* command in PRISM lab computers. For this assignment, you will hand in just one file named **a1.pdf**. When you have completed the assignment, move your **a1.pdf** in a directory (e.g., assignment1), and use the following command to electronically submit your files within that directory:

**% submit 3421 a1 a1.pdf**

You may submit your solution as many times as you wish prior to the submission deadline. Make sure you name your file exactly as stated (including lower/upper case letters). You may check the status of your submission using the command:

**% submit -l 3421 a1**

Once you have submitted, be sure to check that you have submitted the correct version; new or missing files will not be accepted after the due date.