Assignment: The Relational Algebra

COMPSCI 2DB3: Databases–Winter 2024

Deadline: March 7, 2024

Department of Computing and Software
McMaster University

Please read the *Course Outline* for the general policies related to assignments.

Plagiarism is a *serious academic offense* and will be handled accordingly.
All suspicions will be reported to the *Office of Academic Integrity*
(in accordance with the Academic Integrity Policy).

This assignment is an *individual* assignment: do not submit work of others. All parts of your submission *must* be your own work and be based on your own ideas and conclusions. Only *discuss or share* any parts of your submissions with your TA or instructor. You are *responsible for protecting* your work: you are strongly advised to password-protect and lock your electronic devices (e.g., laptop) and to not share your logins with partners or friends!

If you *submit* work, then you are certifying that you are aware of the *Plagiarism and Academic Dishonesty* policy of this course outlined in this section, that you are aware of the Academic Integrity Policy, and that you have completed the submitted work entirely yourself. Furthermore, by submitting work, you agree to automated and manual plagiarism checking of all submitted work.

Late submission policy. Late submissions will receive a late penalty of 20% on the score per day late (with a five hour grace period on the first day, e.g., to deal with technical issues) and submissions five days (or more) past the due date are not accepted. In case of technical issues while submitting, contact the instructor *before* the deadline.

Description

Consider the following relational schema for an online marketplace that consists of the following four relations:

- **user**(*uid*, name, city).

  The **user** relation contains the user name of users and the city in which they live.

- **offer**(*oid*, seller_id, product, price).

  The **offer** relation contains products that are put up for sale by a user (identified via seller_id, a foreign key referencing **user**) for a specified price.

- **sale**(*sid*, customer_id, offer_id, time).

  The **sale** relation contains a record of all sales. In a sale, a customer (identified via customer_id, a foreign key referencing **user**) buys an offer (identified via offer_id, a foreign key referencing **offer**) at a given time (a timestamp that includes both the date and time).

- **sreview**(*sale_id*, score).

  After a sale (identified via sale_id, a foreign key referencing **sale**), the customer can review their experience with the seller by rating their experience (a score between 1 and 10).

- **ureview**(*user_id*, *other_id*, score).

  Each user (identified via user_id, a foreign key referencing **user**) can review their experience with other users (identified via other_id, a foreign key referencing **user**) by rating their experience (a score between 1 and 10).

The requested queries (Relational Algebra)

1. Write a query that returns all users (*uid* from **user**) that have bought products from local sellers (that live in the same city as the user).
2. Write a query that returns all sellers (*seller_id* from **offer**) that are offerign *exactly two* products.
3. Write a query that, for each seller, returns the most expensive offer they have up for offer. Hence, return those rows from **offer** that correspond to a most expensive product that is offered by the seller of that offer.
4. Write a query that returns all user reviews (rows from **ureview**) for which no corresponding sale can be found. Hence, return those reviews in which the user that wrote the review never bought a product from the reviewed seller.
5. Write a query that returns all users (*uid* from **user**) that have reviewed all sellers of which they bought an offer.
6. Not all users review other users in the same manner: some users consider a score of 7 to be high, and other users consider a score of 7 to be low (as they only consider 9 and 10 to be high scores). We want to find those sellers that consistently get high scores from *all users* that review them.

   We say that a user review score $(u, o, s) \in$ **ureview** is a *high score* if the score s is equal to the highest score handed out by the user u among all user review scores of user u in **ureview**.

   Now write a query that returns all reviewed users (the attribute *other_id*) that *only* received reviews with high scores.

Assignment

Write a document in which you answer the above six queries. Hand in a single PDF file in which each answer is clearly demarcated. E.g.,

> **1.** *your first relational algebra query.*
> **2.** *your second relational algebra query.*
> ...
> **6.** *your last relational algebra query.*

Your submission:

1. must include your student number and MacID;
2. must be typed (e.g., generate a PDF via LaTeX or via your favorite word processor): handwritten documents will not be accepted or graded; and
3. all relational algebra queries must use *only* the basic relational algebra (with set semantics) as summarized on the slide "A formal grammar of the relational algebra" (Slide 8 of the slide set "The Relational Algebra").

Submissions that do not follow the above requirements will get a grade of zero.

Grading

Each of the six problems count equally toward the final grade for this assignment. We take the following into account when grading:

1. Is the query correct (does it solve the stated problem)?
2. Are all required columns present?
3. Are no superfluous columns present?