Final Exam

Jelle Hellings

3DB3: Databases – Fall 2021

Department of Computing and Software
McMaster University

Cheating and plagiarism. This exam is an *individual* exam: 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. If you *submit* work, then you are certifying that you have completed the work for that assignment by yourself. By submitting work, you agree to automated and manual plagiarism checking of the submitted work.

Cheating and plagiarism are serious academic offenses. All cases of academic dishonesty will be handled in accordance with the Academic Integrity Policy via the Office of Academic Integrity.

Instructions

- This exam is open-book: you may use the textbook, copies of the lecture slides, material from the tutorials, the assignments and their solutions, the exercises and their solutions.
- Before you start working on the Exam Questions, answer the "Regulatory Questions" on Avenue to Learn.
- All five Exam Questions (Q1–Q5) must be made ***digitally*** and submitted as ***PDF files*** (e.g., written in LaTeX, Microsoft Word, or another word processor—no handwritten scans or tablet drawings).
- Submit each of the five Exam Questions (Q1–Q5) in an individual PDF file. No two questions in the same file!
- Each of the five Exam Questions (Q1—Q5) has their own submit entry on Avenue to Learn. Submit each solution at the right place!
- After finishing the Exam Questions, answer the "Concluding Declarations" on Avenue to Learn.
- We will be flexible with submission times to resolve any technical issues students have—just make sure all your work is submitted at 1:30PM.
- If you have questions during the exam about the exam, then send a direct message (via Microsoft Teams) to Jelle Hellings.

Regulatory Questions

Do you agree with the following statements?

RQ.1. I (the student) understand the above Instructions ("Not Agree" is wrong).

RQ.2. I (the student) am aware that academic integrity requires me to only submit my own answers. I will not seek help for exam questions during the exam, and will only submit answers that are my own work ("Not Agree" is wrong).

RQ.3. I (the student) am aware that during the exam (December 22, 2021 from 9:00AM to 11:30AM) and the hours after the exam (1:30PM), communication about the exam or related topics with other people except the COMPSCI 3DB3 team constitutes academic dishonesty. I commit to not communicate with other people about the exam until 1:30PM ("Not Agree" is wrong).

RQ.4. I (the student) have switched off all communication tools (mobile phones, Facebook, WhatsApp, KakaoTalk, WeChat, Microsoft Teams, Skype, Discord, etc.) and I intend to keep these tools off for the remainder of the exam ("Not Agree" is wrong).

RQ.5. I (the student) agree to automated and manual plagiarism checking of the submitted work ("Not Agree" is wrong).

If you do not agree with these statements, then your exam will not be graded.

Exam Questions

Submissions that do not follow the Instructions will not be graded.

Q1 Theory questions

Q1.1. Explain how comparisons involving SQL NULL-values work within an SQL **WHERE**-clause. You can restrict the answer to only basic comparisons (e.g., $A = B$ with A and B attributes) and Boolean combinations of such comparisons (via the SQL operators **AND**, **OR**, and **NOT**).

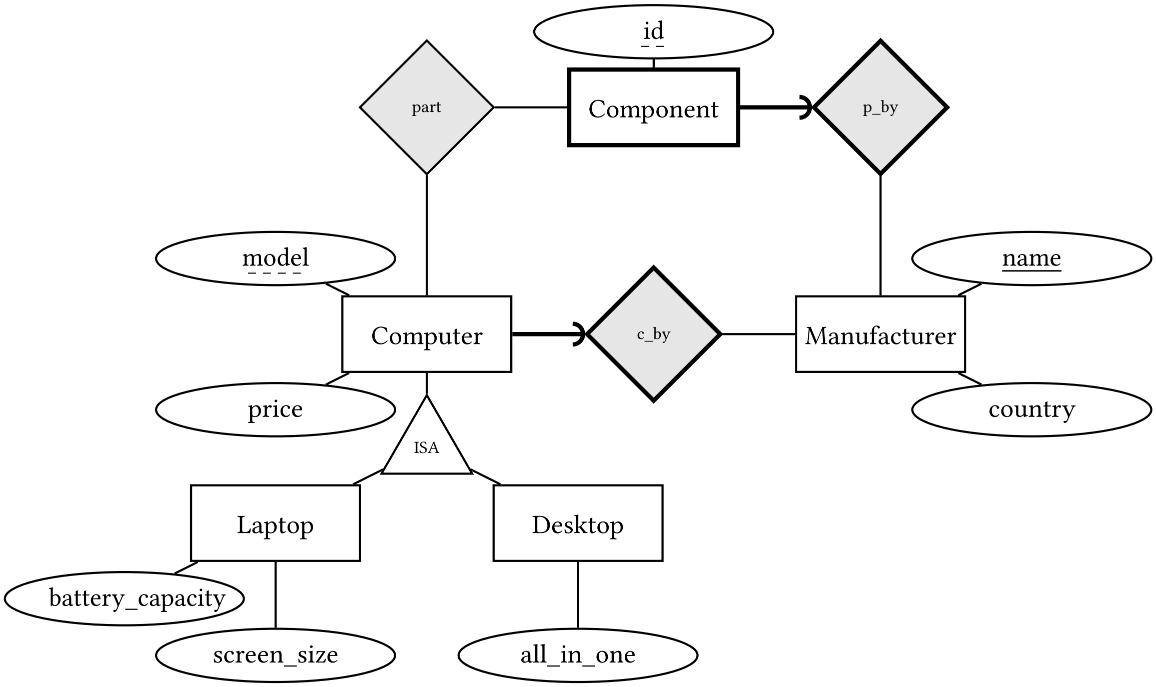
Q1.2. Explain what a *dirty read* is and show how a dirty read can cause *data inconsistencies.*

Q2 The Entity-Relationship Model and the Relational Data Model

Consider the following ER-Diagram that models laptops, desktops, and other computers, and their components. Obviously, not all computers are laptops or desktops (e.g., servers), and laptops and desktops are distinct products.

Q2.1. Translate the ER-diagram to a set of relational schemas (tables). Motivate each of the choices you made during the translation. You do not need to write SQL (e.g., no **CREATE TABLE** statements).

Q2.2. For each table in your solution, mention the primary keys, the foreign keys, the types of each attribute, and any other constraints that apply to your solution.



Q3 SQL Queries

Use the relational schema from the appendix to answer these questions.

Express the following four queries in SQL. Your queries must use the constructs presented on the slides or in the book (we do *not* allow any constructs that are not presented in the slides or in the book).

Q3.1. Return the names of persons (*person* from **Person**) that are not fan of any films.

Q3.2. Return films (*title* from **Film**) that are suitable for all ages: films for which the youngest fan is 60 years younger than their oldest fan.

Q3.3. Return fans (*person* from **FanOf**) that are fans of all films in which the director also acts.

Q3.4. Return, per film (*title* from **Film**) that has *more* fans than actors, the number of fans (column *nfans*) and the number of actors (column *nactors*).

Q4 The Relational Algebra

Use the relational schema from the appendix to answer these questions.

You are free to spell out the relational algebra operators if you have trouble writing them. E.g., instead of $\pi_{\text{p}.age,\text{f}.title}(\sigma_{\text{p}.name=\text{f}.person}(\rho_{\text{p}}(\mathsf{Person}) \times \rho_{\text{f}}(\mathsf{FanOf})))$, you can also write

```
  project_{p.age, f.title}(select_{p.name = f.person}(
                                          rename_p(Person) product rename_f(FanOf))).
```

Express the following three queries in Relational Algebra. Your queries must use the basic relational algebra (with set semantics). Hence, only use relation name atoms, selections (σ), projections (π), unions ($\cup$), intersections ($\cap$), differences ($\setminus$), renames (ρ), cross products ($\times$), and joins and natural joins ($\bowtie$).

Q4.1. Return the films (*title* from **Film**) that have at-least two actors.

Q4.2. Return the films (*title* from **Film**) in which all actors are younger than the youngest fan of that film.

Q4.3. Return the names of directors (*director* from **Film**) that act in every film they directed.

Q5 Dependency Theory, Decomposition, and Normal Forms

Consider the relational schema $\mathbf{r}(A, B, C, D, E, F)$ and the following functional dependencies:

$$\mathfrak{S} = \{AB \longrightarrow DE, C \longrightarrow A, C \longrightarrow E, D \longrightarrow C, E \longrightarrow A\}.$$

Q5.1. Does the functional dependency $CB \longrightarrow D$ follow from $\mathfrak{S}$ (does $\mathfrak{S} \models CB \longrightarrow D$ hold)? Explain why.

Q5.2. Provide a minimal cover and a key for the provided relational schema with respect to the functional dependencies $\mathfrak{S}$.

Q5.3. Is this relational schema in BCNF?

If the relational schema is in BCNF, then explain why.

If the relational schema is not in BCNF, then apply the Decompose-BCNF algorithm to put the relational schema in BCNF. Explain each decomposition step you make, e.g., the functional dependency used to determine a BCNF violation. Provide keys for each of the resulting relational schemas.

Grading

Submissions that do not follow the Instructions will not be graded.

Each of the five parts count toward 20% of the final exam grade.

Appendix

A review website for films holds information on people, films, and the relationships between people and films. The relational schema for this review website consists of the following relations (SQL tables):

- **Person**(name, age, city)
  Each person has a name, an age, and a city in which they live. Each person can be a fan, an actor, or a director.

- **Film**(title, year, director)
  Each film has a unique title and a year in which it is produced. Furthermore, each film has a single director (a person from the **Person** table).

- **PlaysIn**(actor, title)
  This many-to-many relationship relates actors (a person from the **Person** table) to the films (identified by the attribute *title*) in which they play.

- **FanOf**(person, title)
  This many-to-many relationship relates fans (a person from the **Person** table) to the films (identified by the attribute *title*) in which they play.