Example Assignment:
Decomposition and Normal Forms
COMPSCI 2DB3: Databases

Jelle Hellings Holly Koponen

Department of Computing and Software
McMaster University

Description

Part 1: The analysis of a quick-event wizard for a local community

The local community leader came up with the idea of a *quick event wizard* via which users can quickly organize an event (e.g., invite users and order all necessary snacks, drinks, and other products). The community leader already contacted a consultant for an initial sketch of a table that can store all relevant data. The consultant came up with the following relational schema for that table:

event(id, user_id, date, inv_id, inv_confirmed, product, p_price, p_amount).

In this relational schema, each event has a unique identifier *id*. Furthermore, the system keeps track of the user that organizes the event (*user_id*) and the date and time of the event (*date*). The system also keeps track of *all* invited guests. In specific, the systems keeps track, for each invited guest with identifier *inv_id*, whether that guest already confirmed its participation in the event (*inv_confirmed*). Finally, the system keeps tracks of *all* products, e.g., snacks and drinks, that need to be ordered to organize the event (*product*), the price of each of these products (*p_price*) and the amount required of each product (*p_amount*). Next, an example of an instance of this relational schema:

id	user_id	date	inv_id	inv_confirmed	product	p_price	p_amount
1	1	Nov. 3, 3am	2	yes	chips	$2	4
1	1	Nov. 3, 3am	3	no	chips	$2	4
1	1	Nov. 3, 3am	2	yes	cola	$4	8
1	1	Nov. 3, 3am	3	no	cola	$4	8
2	1	Dec. 5, 7pm	2	no	chips	$2	2
2	1	Dec. 5, 7pm	2	no	cola	$4	1

The local community leader is not sure of the quality of this table, but has understood from the consultant that knowing the dependencies that hold on this table will help analyzing the quality of this table. Hence, the local community leader contacted you to determine all dependencies that hold on this table.

Question

1. Provide a minimal cover of *all realistic* non-trivial functional dependencies that hold on the above relational schema. Argue, for each functional dependency, why this functional dependency hold.

Hint: The local community leader only requires a *minimal cover*. Hence, there is no need for trivial functional dependencies and functional dependencies that can be derived from other functional dependencies.

2. Are there any other *non-trivial* dependencies that hold on this table? If so, provide an example of such a dependency and argue why this dependency holds.

   Hint: E.g., multi-valued dependencies, inclusion dependencies, or join dependencies.

Part 2: Refinement of an order-table for a cinema chain

Our familiar local cinema chain owner has evaluated our initial design for the ticket sale and subscription system. Unfortunately, the cinema chain owner concluded that some details and functionality is missing. Hence, the cinema chain owner contacted another consultant to provide a basic design of a relational schema to store all relevant information. Now, the cinema chain owner wants a second opinion on this basic design from an expert. The consultant came up with the following relational schema to store all necessary information:

order(id, screening_time, product,
subscriber_id, sub_start, sub_duration,
film_id, film_length, film_score,
room_id, room_size, room_prop).

In this relational schema, the following order information will be stored:

i. The identifier *id* of the order, the start of the ordered screening (*screening_time*), and all *product*s ordered as part of the order (e.g., the base ticket, special seating, 3D glasses, and so on).

ii. The subscriber that placed the order *subscriber_id*. The system also keeps track of when the subscription of this subscriber started (*sub_start*) and how long the subscriber is subscribed (*sub_duration*).

iii. The film that is shown during the ordered screening (*film_id*) together with the length of the film (*film_length*, which is also the duration of the ordered screening). Furthermore, subscribers can review the film after visiting the screening (*film_score*) and each such score will have a value of *great*, *awful*, and *not-scored*.

iv. The room in which the ordered screening takes place (*room_id*) together with the size of the room (*room_size*) and all technical properties of the room (*room_prop*). Each room can have several technical properties, e.g., with possible values such as *3D*, *Dolby*, and *IMAX*.

Next, an example of an instance of this relational schema (we use shorthand notations for each attribute):

I	St	P	Si	Ss	Sd	Fi	Fl	Fs	Ri	Rs	Rp
1	Nov. 1, 1pm	ticket	1	Oct. 1	31	5	120	great	7	medium	3D
1	Nov. 1, 1pm	ticket	1	Oct. 1	31	5	120	great	7	medium	Dolby
1	Nov. 1, 1pm	3D	1	Oct. 1	31	5	120	great	7	medium	3D
1	Nov. 1, 1pm	3D	1	Oct. 1	31	5	120	great	7	medium	Dolby
2	Nov. 1, 1pm	ticket	2	Oct. 3	29	5	120	awful	7	medium	3D
2	Nov. 1, 1pm	ticket	2	Oct. 3	29	5	120	awful	7	medium	Dolby
2	Nov. 1, 1pm	3D	2	Oct. 3	29	5	120	awful	7	medium	3D
2	Nov. 1, 1pm	3D	2	Oct. 3	29	5	120	awful	7	medium	Dolby
3	Nov. 7, 2pm	ticket	2	Oct. 3	29	9	99	not-scored	3	large	IMAX
3	Nov. 7, 2pm	IMAX	2	Oct. 3	29	9	99	not-scored	3	large	IMAX
3	Nov. 7, 2pm	ticket	2	Oct. 3	29	9	99	not-scored	3	large	4D
3	Nov. 7, 2pm	IMAX	2	Oct. 3	29	9	99	not-scored	3	large	4D

According to the consultant, the set of attributes "*id, product, room_prop*" is a key, the following additional functional dependencies hold:

$$
\begin{aligned}
\text{id} &\longrightarrow \text{screening\_time, subscriber\_id, sub\_start, sub\_duration;}\\
\text{id} &\longrightarrow \text{film\_id, film\_length, film\_score, room\_id, room\_size;}\\
\text{subscriber\_id} &\longrightarrow \text{subscriber\_id, sub\_start, sub\_duration;}\\
\text{sub\_start} &\longrightarrow \text{sub\_duration;}\\
\text{sub\_duration} &\longrightarrow \text{sub\_start;}\\
\text{screening\_time, room\_id} &\longrightarrow \text{film\_id;}\\
\text{film\_id} &\longrightarrow \text{film\_length;}\\
\text{film\_id, subscriber\_id} &\longrightarrow \text{film\_score; and}\\
\text{room\_id} &\longrightarrow \text{room\_size.}
\end{aligned}
$$

Question

3. Is the relational schema **Order** in 3NF?

   If so, then explain why **Order** is in 3NF.

   Otherwise, decompose the schema using the 3NF Synthesis algorithm (Decompose-3NF) and document each step you make while applying the algorithm. Provide the functional dependencies that hold in each relational schema in your resulting decomposition (a minimal cover suffices). Explain whether this decomposition is lossless-join and whether it is dependency-preserving (with respect to the original functional dependencies). Finally, decompose the example dataset according to the relational schema obtained from the decomposition algorithm.

4. Is the relational schema **Order** in BCNF?

   If so, then explain why **Order** is in BCNF.

   Otherwise, decompose the schema using the BCNF Decomposition algorithm (Decompose-BCNF) and document each step you make while applying the algorithm. Provide the functional dependencies that hold in each relational schema in your resulting decomposition (a minimal cover suffices). Explain whether this decomposition is lossless-join and whether it is dependency-preserving (with respect to the original functional dependencies). Finally, decompose the example dataset according to the relational schema obtained from the decomposition algorithm.

5. According to the consultant, the following multi-valued dependencies also hold:

$$\text{ID} \twoheadrightarrow \text{product}; \text{ and}$$
$$\text{ID} \twoheadrightarrow \text{room\_prop},$$

   in which ID = {id, screening_time, subscriber_id, sub_start, sub_duration, film_id, film_length, film_score, room_id, room_size}. Is the relational schema **Order** in 4NF?

   If so, then explain why **Order** is in 4NF.

   Otherwise, decompose the schema using the 4NF Decomposition algorithm (Decompose-4NF) and document each step you make while applying the algorithm. Provide the functional dependencies that hold in each relational schema in your resulting decomposition (a minimal cover suffices). Explain whether this decomposition is lossless-join and whether it is dependency-preserving (with respect to the original functional dependencies). Finally, decompose the example dataset according to the relational schema obtained from the decomposition algorithm.

6. Does any of the above three decompositions of **Order** resolve all design issues of **Order**? If so, explain which decomposition(s) resolve all design issues. Else, provide an example of a design issue that was not resolved by decomposition.

Assignment

The goal of the assignment is to help out the local community leader and the local cinema chain owner. To do so, answer Questions 1–6.