UNIVERSITY OF
GOTHENBURG

CHALMERS
UNIVERSITY OF TECHNOLOGY

Exam

DIT032 / DAT335 – Data Management

Wednesday, August 21st, 2019 08:30 - 12:30

Examiner:
Philipp Leitner

Contact Persons During Exam:
Philipp Leitner (+46 733 05 69 14)
Joel Scheuner (+46 733 42 41 26)

Allowed Aides:
None except English dictionary (non-electronic), pen/pencil, ruler, and eraser.
Check the back of your exam papers for additional help in the appendix.

Results:
Exam results will be made available no later than in 15 working days through Ladok.

Grade Limits:
For GU students: 0 – 49 pts: U, 50 – 84 pts: G, 85+ pts: VG
For Chalmers students: 0 – 49 pts: U, 50 – 69 pts: 3, 70 – 84 pts: 4, 85+ pts: 5

Review:
The exam review will take place Friday, September 13th (10:00 - 11:00) in Philipp Leitner's office (room 411, Jupiter building, 4th floor). Please refer to Canvas for potential updates.

Task 1 – Theory and Understanding (25 pts)

(every question can and should be answered in a few sentences, plus an example if asked for)

Q1.1: Sketch the ANSI/SPARC Three-Schema database architecture. *2 pts*

Q1.2: What is a “weak entity” in the Entity Relationship model? What is special about them (e.g., in terms of key requirements)? When would you typically use a weak entity? *4 pts*

Q1.3: Explain the concept of a “full table scan” in relational databases. Give a short example. *2 pts*

Q1.4: In a relational database, we typically want to avoid redundancies. Why? Which redundancies cannot be avoided? *3 pts*

Q1.5: What are transactions in a database system? What problem do transactions solve? Describe how transactions are implemented in the SQL standard. *5 pts*

Q1.6: What is replication? What problem does it solve? What does it specifically not help with? *3 pts*

Q1.7: Describe the CAP theorem. What dimension(s) of the CAP theorem do relational databases usually emphasize? *4 pts*

Q1.8: Briefly describe the relationship between the JSON and YAML data representation formats. *2 pts*

Task 2 – EER Diagrams (23 pts)

Consider the following simplified domain description of the social media platform Twitter. Model the described domain using an EER diagram with the notation we used in the course (find a cheat sheet in the appendix of your exam papers). Use the 1,N,M notation for describing cardinalities rather than the min-max notation.

Twitter:
Our database needs to store Twitter users, tweets, and direct message conversations. Users have an unique username, a full name (which is composed of first and last name), and a registration date. Users subscribe to updates of other users by “following ” them. Any user can follow any number of others, and any user can have an arbitrary number of followers. We need to store the timestamp representing since when a user follows another user. Users should also have an attribute that captures how many other users they are following.

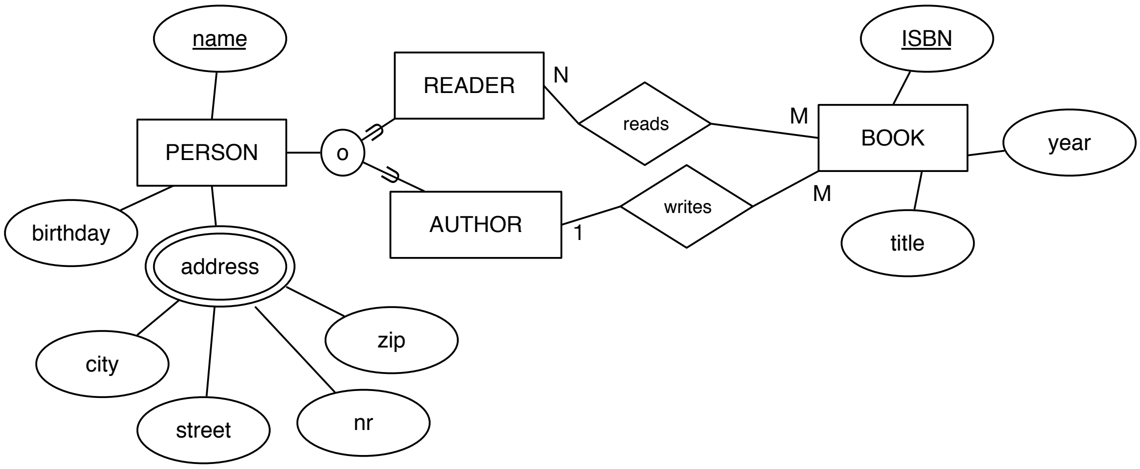
Users may send an arbitrary number of tweets (short messages). A tweet consists of a globally unique identifier, a text, and the date when the tweet has been sent. A tweet may also tag zero to many users. Finally, a tweet may be a retweet of a previous message. In that case the tweet needs to refer back to the original message. However, a single tweet may only retweet (at most) one previous tweet. For each retweet, we need to store the time stamp of the retweet.

There are two special types of tweets. Firstly, replies are tweets sent in response to a previous tweet. In addition to all properties of normal tweets, replies have a thread number and point to exactly one previous tweet. Secondly, media tweets are defined as tweets that have one or more media items (e.g., pictures or videos) embedded. Tweets can at the same time be replies and/or media tweets (or neither, of course). For media items we need to store a globally unique identifier and the URL of the media item.

In addition to publicly viewable tweets, users may also have private conversations. A conversation is always between exactly two users (*note: consider how to map this requirement using the 1/N/M notation!*), is identified through a globally unique ID, and points to both users involved in the conversation. Further, a conversation contains one to many messages. Messages are identified through a running number that is only unique in the context of a specific conversation. Further, messages contain the message text, and they may optionally also embed one or multiple media items.

Task 3 – Mapping EER to the Relational Model (12 pts)

Consider the EER model below. Construct a relational model that represents this domain. Select primary keys and introduce foreign keys as necessary. Use the notation that we used in the course to indicate relations, attributes, primary keys, and foreign keys (see Task 4 for an example of the required notation).



Task 4 – Relational Algebra (20 pts)

Relational Model:
LIBRARY($\underline{\text{name}}$, location)

SHELF($\underline{\text{library}}$, $\underline{\text{shelf\_nr}}$, topic)
library $\rightarrow$ LIBRARY.name

BOOK($\underline{\text{isbn}}$, title, author)

BOOK_PLACED($\underline{\text{book}}$, $\underline{\text{library}}$, $\underline{\text{shelf}}$, since, borrowed)
book $\rightarrow$ BOOK.isbn
$\{$library,shelf$\} \rightarrow \{$SHELF.library,SHELF.shelf_nr$\}$

Note: Library and shelf_nr in BOOK_PLACED are a composed foreign key pointing at the composed primary key of SHELF. The attribute “since” is a timestamp indicating when this book was placed. “Borrowed” is a boolean flag indicating if the book is currently lended out.

Given this relational model, write relational algebra statements that exactly represent the following queries. Use the mathematical notation from the course (for the correct notation you can again refer to the appendix).

Queries:

Q4.1: Find all currently lended out books from the library “GU lib”.

Q4.2: Find the titles of all books in libraries located in Gothenburg.

Q4.3: Return a list of topics and how many books are placed in total in shelves of this topic (in any library). Topics with no books should still be included.

Q4.4: Find the titles of all books that are not placed in the library “GU lib”.

Task 5 – SQL (20 pts)

Given the same relational model as for Task 4, write the following SQL statements or queries.

Statements:

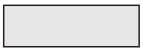
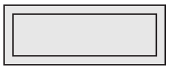
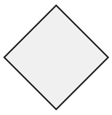
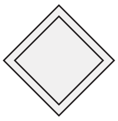
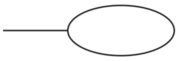
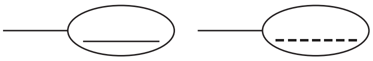
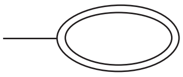
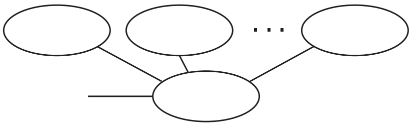
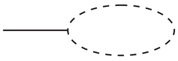
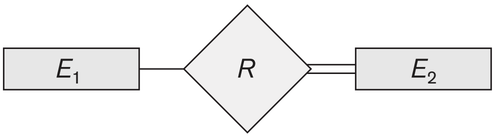
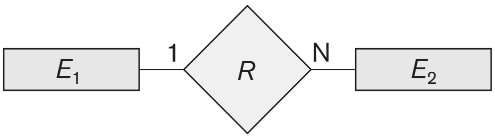
Q5.1: Create the table `SHELF`. Assume that the table `LIBRARY` with the schema defined in Task 4 already exists. Select suitable data types, and define primary and foreign keys as in the relational model.

Q5.2: Find the titles and authors of all books that are currently not being borrowed, and which have been placed before 2019-01-01.

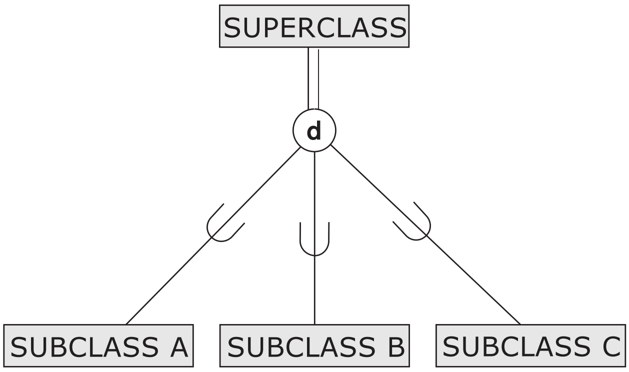
Q5.3: Count how many libraries there are in each location.

Q5.4: Find the ISBN numbers of all books that are not placed under the topic “Computer Science” in any library.

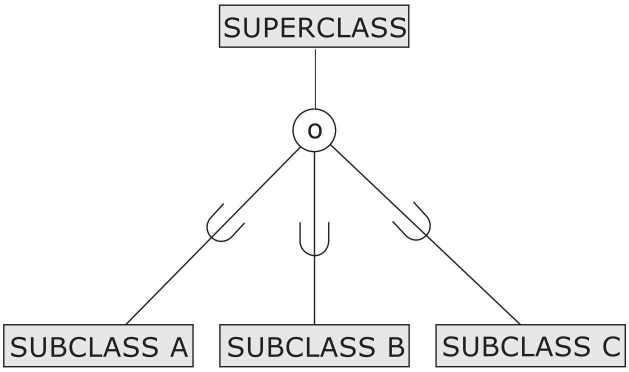
Appendix: Notation Guidelines for EER and RA

Symbol	**Meaning**
	Entity
	Weak Entity
	Relationship
	Indentifying Relationship
	Attribute
	Key Attribute / Dashed Underline for Partial Key
	Multivalued Attribute
	Composite Attribute
	Derived Attribute
	Total Participation of E_2 in R
	Cardinality Ratio 1: N for E_1 : E_2 in R

Total Disjoint Specialization



Partial Overlapping Specialization



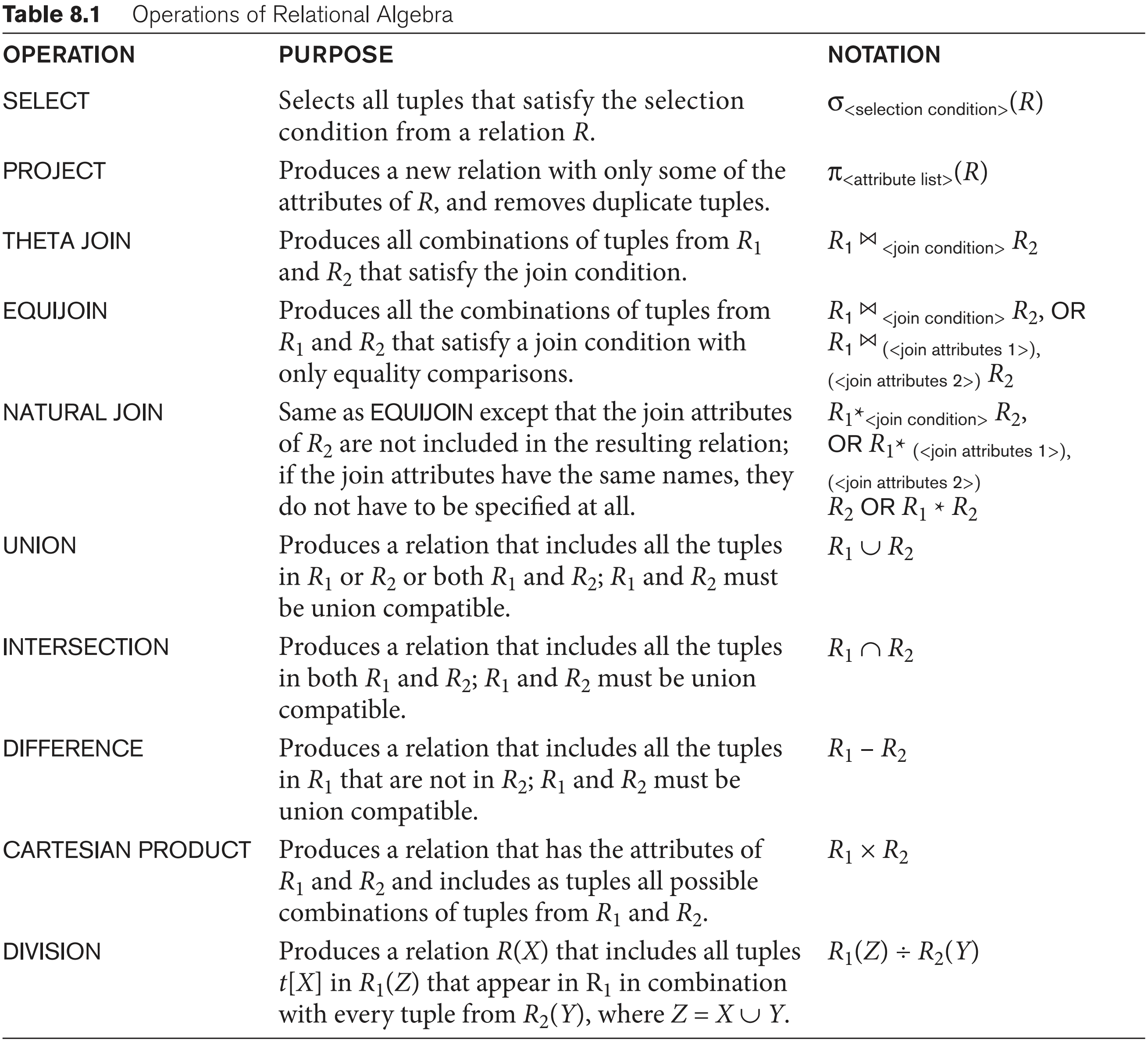


Table 8.1 Operations of Relational Algebra

OPERATION	PURPOSE	NOTATION
SELECT	Selects all tuples that satisfy the selection condition from a relation R.	$\sigma_{\text{<selection condition>}}(R)$
PROJECT	Produces a new relation with only some of the attributes of R, and removes duplicate tuples.	$\pi_{\text{<attribute list>}}(R)$
THETA JOIN	Produces all combinations of tuples from R_1 and R_2 that satisfy the join condition.	$R_1 \bowtie_{\text{<join condition>}} R_2$
EQUIJOIN	Produces all the combinations of tuples from R_1 and R_2 that satisfy a join condition with only equality comparisons.	$R_1 \bowtie_{\text{<join condition>}} R_2$, OR $R_1 \bowtie_{\text{(<join attributes 1>),}}$ $_{\text{(<join attributes 2>)}} R_2$
NATURAL JOIN	Same as EQUIJOIN except that the join attributes of R_2 are not included in the resulting relation; if the join attributes have the same names, they do not have to be specified at all.	$R_1 *_{\text{<join condition>}} R_2$, OR $R_1 *_{\text{(<join attributes 1>),}}$ $_{\text{(<join attributes 2>)}}$ R_2 OR $R_1 * R_2$
UNION	Produces a relation that includes all the tuples in R_1 or R_2 or both R_1 and R_2; R_1 and R_2 must be union compatible.	$R_1 \cup R_2$
INTERSECTION	Produces a relation that includes all the tuples in both R_1 and R_2; R_1 and R_2 must be union compatible.	$R_1 \cap R_2$
DIFFERENCE	Produces a relation that includes all the tuples in R_1 that are not in R_2; R_1 and R_2 must be union compatible.	$R_1 - R_2$
CARTESIAN PRODUCT	Produces a relation that has the attributes of R_1 and R_2 and includes as tuples all possible combinations of tuples from R_1 and R_2.	$R_1 \times R_2$
DIVISION	Produces a relation $R(X)$ that includes all tuples $t[X]$ in $R_1(Z)$ that appear in R$_1$ in combination with every tuple from $R_2(Y)$, where $Z = X \cup Y$.	$R_1(Z) \div R_2(Y)$

<grouping>$\mathscr{F}$<functions>(R)

whereas <functions> is a list of

```
[MIN|MAX|AVERAGE|SUM|COUNT] <attribute>
```