UNIVERSITY OF
GOTHENBURG

CHALMERS
UNIVERSITY OF TECHNOLOGY

Exam

DIT032 / DIT033 / DAT335 – Data Management

Thursday, June 9, 2022, 08:30 - 12:30

Examiner:
Philipp Leitner

Contact Persons During Exam:
Nayla Nasir (+46 31 772 1986)
(visitations around 9:30 and 11:00)

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 – 49.9 pts: U, 50 – 69.9 pts: 3, 70–84.9 pts: 4, 85+ pts: 5
For Chalmers students: 0 – 49.9 pts: U, 50 – 69.9 pts: 3, 70 – 84.9 pts: 4, 85+ pts: 5

Task 1 – Theory and Understanding (22 pts)

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

Q1.1: What is a "weak entity" in the Entity Relationship model? Give an example of a weak entity. What is special about weak entity (e.g., in terms of key requirements)? (3 pts)

Q1.2: Describe, in your own words, the concept of semi-structured data. How does it differ from structured and unstructured data? (5 pts)

Q1.3: Explain the concept of a "full table scan" in relational databases. Give a short example. (2 pts)

Q1.4: What problem do views in SQL solve? Which different types of views are there? (3 pts)

Q1.5: Name and briefly explain the four "ACID" properties of database transactions. (4 pts)

Q1.6: What is sharding? What problem does it solve? Illustrate with an example how sharding utilizes mapping function to map content across different shards (5 points).

Task 2 – EER Diagrams (20 pts)

Consider the following excerpt of the domain description for a conference database. 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.

Conference Management:
The database needs to keep track of conferences. Conferences have an unique name, a location, a duration, and an edition (e.g., 2nd edition). The duration is specified through a begin and end date. We also want to track how many conferences there are in total in the system.

Conferences are attended by delegates (i.e., people attending the conference). Delegates can attend multiple conferences, and clearly a conference can have many delegates attending. For each delegate, we need to store when they registered for each conference ("reg_date"). For each delegate, we further need to store their registration number (which is unique for each delegate), their name, country, and date of birth.

Conferences further consist of tracks. Tracks have a name which is unique for a conference (but not necessarily unique overall), a date, start time, and end time. Every track has exactly one track chair, who is a special delegate responsible for coordinating the track. For delegates that are also track chairs, we need to save the organisation that they represent in the conference (this is not needed for regular delegates). A track chair can chair multiple tracks, but they need to chair at least one to be considered a track chair.

Finally, tracks consist of presentations. For each presentation, we need to save the title, which is unique in a track (but, again, not necessarily unique overall), and the start time. Each presentation is given by exactly one delegate (the presenter), but there may be delegates who are not giving any presentations. Each delegate may give at most one presentation.

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

Consider the EER model for Task 2. 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:
CITY(name)
PERSON(ssn, fname, lname, dob, location)
PERSON.location is a foreign key pointing at CITY.name
LIBRARY(id, name, location)
LIBRARY.location is a foreign key pointing at CITY.name
BOOK(isbn, title, page_count, cover)
BOOK_LIB(book, library)
BOOK_LIB.book is a foreign key pointing at BOOK.isbn
BOOK_LIB.library is a foreign key pointing at LIBRARY.id

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 the first names of all persons with the last name "Josh". The resulting relation should have a single attribute called `Josh_names`.

Q4.2: List ISBN and titles for all the books and the name of the libraries that hold them. Books not available in any library should still be contained in the list (just without any library information).

Q4.3: For each book with the hard cover (`cover = 'hc'`), return the title of the book, its page_count as well as the name and location of the library that holds it.

Q4.4: Count how many books there are in each library.

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 `PERSON` with schema defined in Task 4. Add additional constraints that ensure that persons are 18 years or older (`dob` smaller than 2004-06-09).

Q5.2: Return the first name, last name, and location of all persons living in a city ending with "köping".

Q5.3: Find all book titles and the locations (cities) that they are available in. Books not available in any library should not appear in the result table.

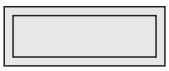
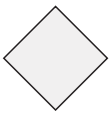
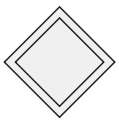
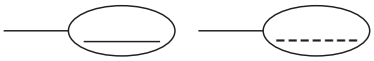
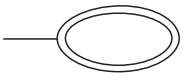
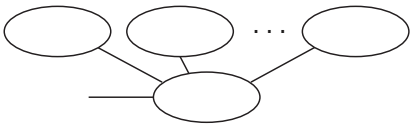
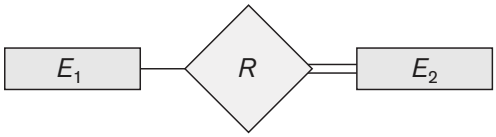
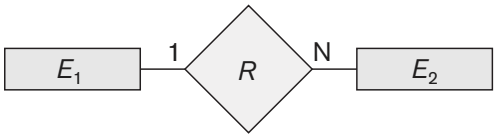
Q5.4: Count how many soft-cover books (`cover = 'sc'`) there are in the library at the Gothenburg location.

Task 6 – Data representation (6 pts)

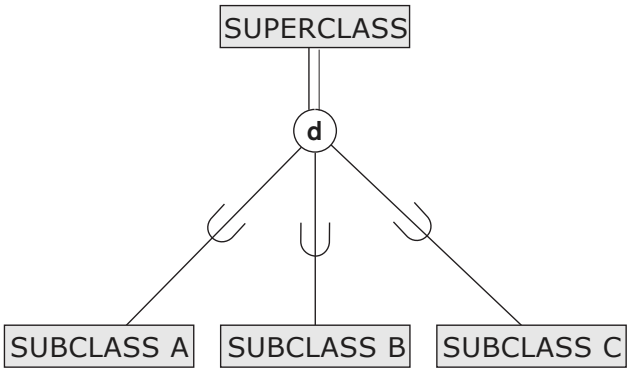
```
<?xml version="1.0" encoding="UTF-8" ?>
<conference>
<tracks>
   <presentation id="1" title=Databases for Beginners
   <presentation id="1" id="2" title="Advanced Databases">
   </presentation>
</tracks>
```

This minimal XML snippet is not well-formed (i.e., it violates the basic syntactical rules of XML). Name and briefly explain **at least 3** reasons for this document not being well-formed.

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

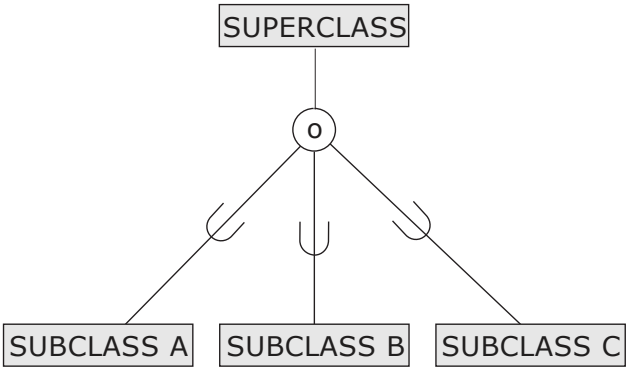


Table 8.1 Operations of Relational Algebra

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 $R_1 \bowtie_{\text{(<join attributes 1>), (<join attributes 2>)}} R_2$
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 $R_1 *_{\text{(<join attributes 1>), (<join attributes 2>)}} R_2$ 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>