



UNIVERSITY OF
GOTHENBURG

CHALMERS
UNIVERSITY OF TECHNOLOGY

Exam

DIT032 / DAT335 – Data Management

Monday, March 16th, 2020 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)

(teacher's visits around 09:30 and 10:30)

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:

Date and time of the exam review will be announced via Canvas.

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: Briefly describe the idea behind weak entity types in (E)ER diagrams. What are they used for? *3 pts*

Q1.2: What are functional dependencies in a relational model? Provide an example of a relational model (using the notation used in the course, see also Task 3) which contains (at least) one redundant functional dependency. *4 pts*

Q1.3: What are transactions in a relational database? Name and describe the three transaction boundaries (transaction primitives) we discussed. *4 pts*

Q1.4: What is a database index in a relational database? What problem do indices solve? Discuss based on an example. *4 pts*

Q1.5: Describe and contrast structured, semi-structured, and unstructured data. *3 pts*

Q1.6: Explain the basic idea behind Map/Reduce using an example. How can you use Map/Reduce in a MongoDB database? *4 pts*

Task 2 – EER Diagrams (18 pts)

Consider the following excerpt of the domain description for the database of an online teaching platform (similar to Canvas). 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. If *and only if* something is not specified, make a reasonable assumption and note it down in plain text.

Online Teaching Platform:

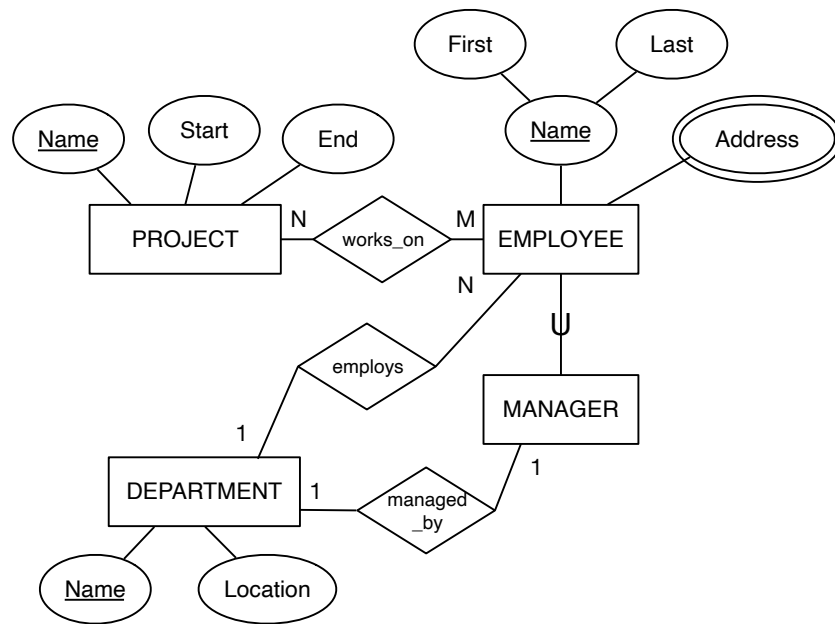
The database needs to keep track of courses, assignments, and (different types of) users. For every user, we need to store a login name (which is also the unique identifier of users), a password hash, and previous login attempts as a list of dates. We need to distinguish three types of users: teachers, students, and teaching assistants (which are a special type of student). For teaching assistants we need to also store their salary rate. Note that in special cases a teacher may also be a student.

Courses have a unique short name, an equally unique full name, and a syllabus. Every course is taught by one or multiple teachers, and taken by multiple students (but there may also be courses that are not taken by any student). For each student taking a course, we need to keep track of the student's grade. Our database also should offer a way to retrieve how many students are taking a course.

Every course can (optionally) have assignments. Assignments are identified through a number (which is only unique for a given course), and we also need to store the assignment text and due date for each assignment. Every assignment is graded by exactly one teaching assistant. Most teaching assistants will be grading multiple assignments (but every teaching assistant grades at least one).

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:

STUDENT(firstname, lastname, semester)

COURSE(shortname, fullname, start, end)

TEACHER(id, name, salary, course)

course → COURSE.shortname

ASSIGNMENT(number, course, solution, grade, student_fname, student_lname)

{student_fname, student_lname} → {STUDENT.firstname, STUDENT.lastname}

(course is a foreign key pointing at COURSE.shortname,
student_fname and student_lname together point at the composed primary key of
STUDENT)

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 short and full name of all courses starting after “2020-01-01” and ending before “2020-05-01”.

Q4.2: List all assignments of students currently studying in their third semester (i.e., *semester* = 3).

Q4.3: For the course with the short name “DAT335”, count how many students had the grade 5 per assignment.

Q4.4: List all teachers and all information about the course they are teaching. Teachers not currently teaching any courses should still be contained in the list.

Task 5 – SQL (20 pts)

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

Statements:

Q5.1: Return grades of all assignments by the student with the first name “Philipp” and last name “Leitner”.

Q5.2: Return all courses starting between “2020-01-01” and “2020-03-01”, all assignments for these courses, and the teachers teaching these courses, ordered first by the course’s starting date and then by assignment grade.


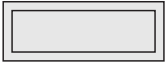
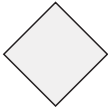
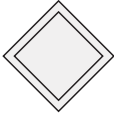

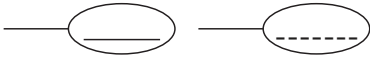
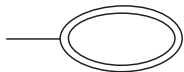
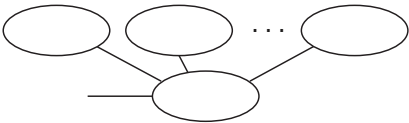

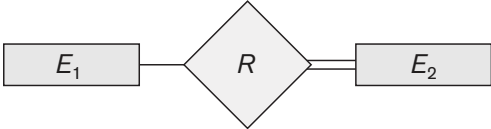
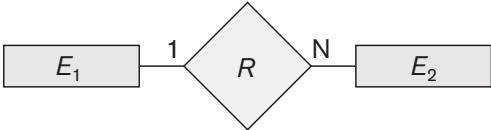
Q5.3: Return all attributes of all pairs of students with the same last name.

Q5.4: Return the names of all teachers with above-average salary.

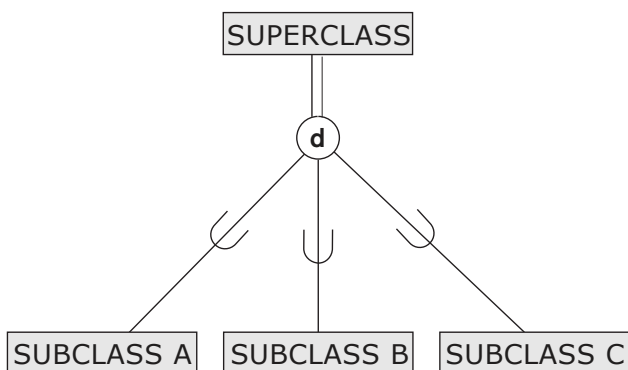
Task 6 – XML (8 pts)

Refer back to the domain model used in Task 3 (projects and employees). Write an example XML document representing one example department, two employees, and the department's manager. All attributes of these entity types should be used at least once. Decide yourself which attributes to model as subelements, and which as attributes. Use arbitrary data as attribute values. Take care that your file is well-formed XML. However, you do not need to add an XML header.

Appendix: Notation Guidelines for EER and RA

Symbol	Meaning
	Entity
	Weak Entity
	Relationship
	Identifying 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

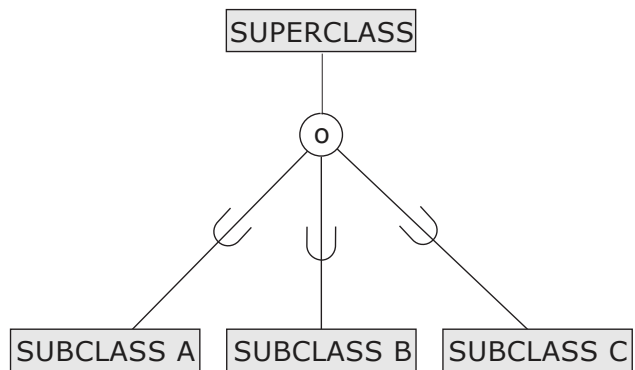


Table 8.1 Operations of Relational Algebra

OPERATION	PURPOSE	NOTATION
SELECT	Selects all tuples that satisfy the selection condition from a relation R .	$\sigma_{\langle \text{selection condition} \rangle}(R)$
PROJECT	Produces a new relation with only some of the attributes of R , and removes duplicate tuples.	$\pi_{\langle \text{attribute list} \rangle}(R)$
THETA JOIN	Produces all combinations of tuples from R_1 and R_2 that satisfy the join condition.	$R_1 \bowtie_{\langle \text{join condition} \rangle} R_2$
EQUIJOIN	Produces all the combinations of tuples from R_1 and R_2 that satisfy a join condition with only equality comparisons.	$R_1 \bowtie_{\langle \text{join condition} \rangle} R_2$, OR $R_1 \bowtie_{(\langle \text{join attributes 1} \rangle, \langle \text{join attributes 2} \rangle)} R_2$
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.	$R_1 \star_{\langle \text{join condition} \rangle} R_2$, OR $R_1 \star_{(\langle \text{join attributes 1} \rangle, \langle \text{join attributes 2} \rangle)} R_2$
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 $Z = X \cup Y$.	$R_1(Z) \div R_2(Y)$

$\langle \text{grouping} \rangle \mathcal{F} \langle \text{functions} \rangle (R)$

whereas $\langle \text{functions} \rangle$ is a list of

[MIN | MAX | AVERAGE | SUM | COUNT] $\langle \text{attribute} \rangle$