Exam Database Systems – Spring 2018

Please read the following instructions carefully:

- Please fill in your personal information on the first sheet.
- This exam has 7 exercises and 19 pages. Make sure your copy contains all exercises.
- Whenever possible, write your answers directly in the provided fields. If you run out of space, you can use the additional sheets of paper attached at the end of the exam copy.
- Read the text of each exercise carefully before solving it.
- If you get stuck on an exercise, proceed with another one, and come back later.
- If your solution is based on an assumption that you have made, please add a comment for clarification.
- During the exam, no aids are allowed, i.e., you are not allowed to consult any books, notes, electronic devices, talk to someone, etc.
- The time limit is 2 hours.
- The exam is passed with 50 points.
- Please use a readable handwriting and try to present your solutions neatly. Make sure that text that is not part of your solution is crossed out.
- Good luck!

Name: ______________________________ **Study No.:** ____________

Results

To be filled in during grading

Exercise	1	2	3	4	5	6	7	$\sum$
Max. Points	15	30	10	7	15	10	13	100
Achieved								

Comments	

1 ER Modeling (15 points)

1.1 ER Modeling

We would like to design a lightweight database for a hospital and capture the following information:

- Each department has at least one patient room and at least one doctor. For each department, we store the unique department name, address, and phone number.
- Each patient room is associated with exactly one department, and can take at most 4 patients. For each patient room, we store a phone number, and it is uniquely identified by a room number.
- Each doctor works for exactly one department, but can treat an arbitrary number of patients. For each doctor, we store employID, name, and rank.
- Each patient in the database is associated with exactly one doctor, and stays in at most one patient room. For each patient, we store patientID, name, and gender.

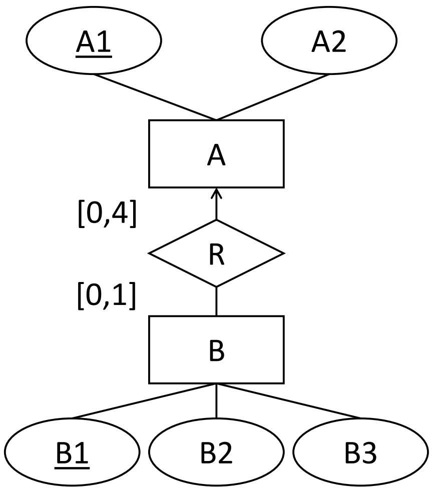
Model all the information and do not add more (in particular, no additional attributes).
Please identify and mark **primary keys** and add information about cardinality ratios (**Chen notation**) and cardinality limits (**[min,max] notation**). Use the notation introduced in the course.

Simplications:

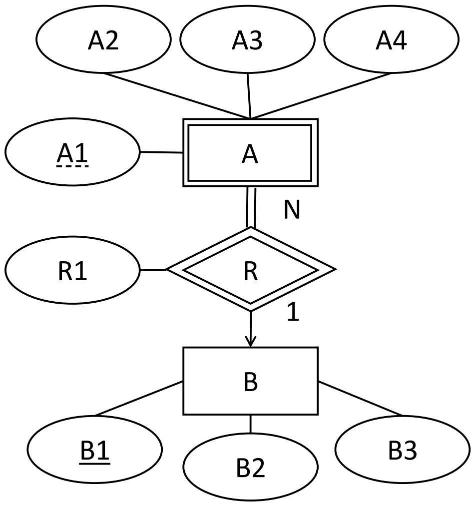
- It is not necessary to indicate data types.
- Do not use advanced concepts such as generalization or weak entity types.

1.2 Relational Modeling

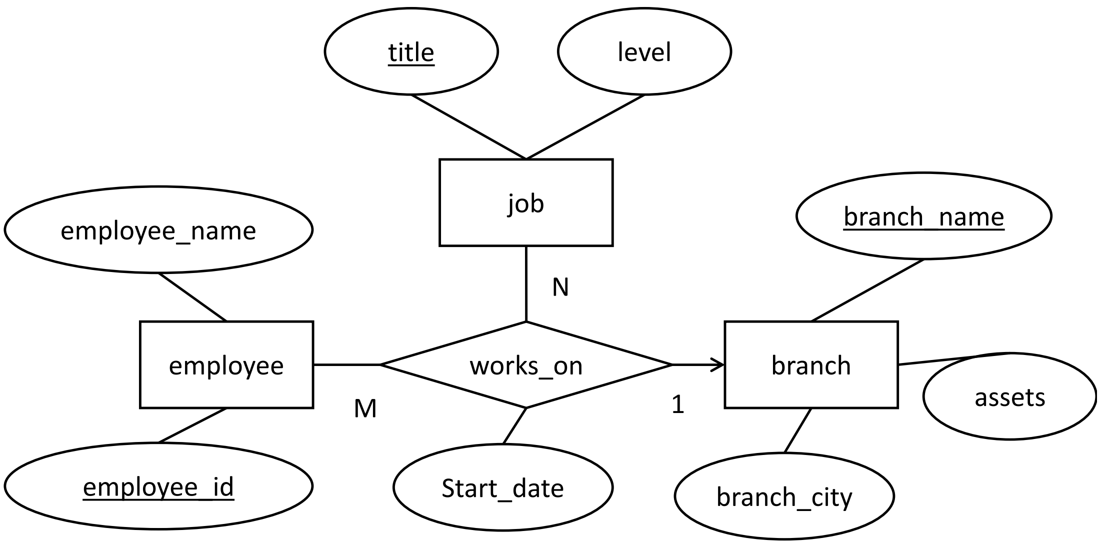
Consider the following three ER diagrams:



(a) ERD 1



(b) ERD 2



(c) ERD 3

For each of these ER diagrams, please create an appropriate **set of relations**. Please aim for a **minimal** representation that avoids storing redundant information or not strictly required information (in particular, no additional attributes!). Identify and mark **primary keys** in your relations. Likewise, identify, note down, and mark **foreign keys**. It is not necessary to indicate data types.

Relations for ERD 1

Relations for ERD 2

Relations for ERD 3

2 Query languages (30 points)

Consider the following relations capturing information about deliverymen, restaurants, cuisines, and delivery orders:

- `restaurant(rid, name, street, number, postalCode, city)`
- `cuisine(cid, name, type, rid → restaurant)`
- `deliveryman(pid, CPR, name, joinDate)`
- `dorder(pid → deliveryman, cid → cuisine, orderTime, orderDate, tip)`

where the primary keys are underlined and the foreign keys denoted by arrows. Only integer and string are used as data types.

For each of the following, write a query (a single statement) in the specified query language (SQL or relational algebra) that produces the requested information. Note that your solution should not output additional information.

1. **Relational algebra**:
   Find all the deliverymen who joined as deliveryman since the beginning of 2010. For each such deliveryman, list his/her pid, CPR number, and name.
   Hint: you can use "DD.MM.YYYY" as date format.

2. **Relational algebra**:
   Find the names of all cuisines and all corresponding restaurants with orders placed after 20:00.

3. **SQL**:
   List the unique names of all deliverymen that have delivered orders of any cuisine of restaurant "Nanking Grill"?

4. **SQL**:
   Find all the cuisines that have never been ordered. For each such cuisine, find the names of all restaurants offering it.
   Hint: It is sufficient to output cuisine-restaurant combinations.

5. **SQL**:
   What are the names of the cuisines that have been ordered the most? Limit your answer to the top 5.

6. **Relational algebra**:
   Find the IDs of all deliverymen who delivered orders of cuisines from any restaurant with postalCode '9000'.

7. **SQL**:
   How many deliverymen have delivered orders of the cuisine “open sandwich”?

3 Normal Forms (10 points)

1. Given the following relation with its functional dependencies, determine the set of **candidate keys**. Then determine which normal forms are fulfilled (restrict your consideration to **2NF**, **3NF**, and **BCNF**). You may assume that **1NF** is always fulfilled. **Explain** your answer with respect to **each** normal form (not just the highest one that is fulfilled)!

(a) $\mathcal{R} = \{A, B, C\}$
$AB \rightarrow C$
$BC \rightarrow A$

(b) $\mathcal{R} = \{A, B, C, D, E, F\}$
$AB \rightarrow CDEF$
$CD \rightarrow B$
$E \rightarrow F$

4 Lossless Decomposition and Dependency Preservation (7 points)

Given the following relational schema

$\mathcal{R}$: {[attr1, attr2, attr3, attr4, attr5, attr6]}

Assume the following FDs (and only these) hold on this relation:

- { attr2, attr3 } → { attr6 }
- { attr1 } → { attr2, attr3, attr4, attr5 }
- { attr4 } → { attr5 }

To avoid redundancy, the schema is decomposed into two parts $\mathcal{R}_1$ und $\mathcal{R}_2$ with

$\mathcal{R}_1$: {[attr1, attr2, attr3, attr4, attr5]}
$\mathcal{R}_2$: {[attr4, attr6]}

Please answer the following questions:

1. Is the decomposition lossless? Please explain your answer.

2. Is the decomposition dependency preserving? Please explain your answer.

5 Serializability (15 points)

Consider two schedules S_1 and S_2:

$$S_1 := r_1(a) \rightarrow r_2(a) \rightarrow r_1(b) \rightarrow r_3(b) \rightarrow w_1(b) \rightarrow w_2(a) \rightarrow w_3(b) \rightarrow r_2(b) \rightarrow c_1 \rightarrow c_2 \rightarrow c_3$$

$$S_2 := r_4(y) \rightarrow r_5(y) \rightarrow w_5(x) \rightarrow w_2(y) \rightarrow r_1(x) \rightarrow c_2 \rightarrow c_1 \rightarrow r_3(y) \rightarrow c_3 \rightarrow w_3(z) \rightarrow c_5 \rightarrow w_4(z) \rightarrow c_4$$

where

- $r_1(x)$ encodes that transaction T_1 reads data item x (w represents write access)
- c_1 encodes that transaction T_1 commits
- $\rightarrow$ encodes the order in which the operations are executed

For both schedules (S_1 and S_2), please create a **conflict graph** and note down whether the schedule is conflict **serializable**. If it is, give an example of a **conflict equivalent serial schedule**. If it is not, explain your answer.
Hint: Annotating edges with additional information is not necessary.

S_1:

S_2:

6 Query Optimization (10 points)

Consider the following abstract relations:

- $\texttt{R1(}\underline{\texttt{R1A, R1B}}\texttt{, R1C, R1D} \rightarrow \texttt{R4.R4A, R1E)}$
- $\texttt{R2(}\underline{\texttt{R2A}}\texttt{, R2B, R2C)}$
- $\texttt{R3(}\underline{\texttt{R3A, R3B, R3C} \rightarrow \texttt{R2.R2A}}\texttt{, R3D, R3E)}$
  $\{\ \texttt{R3.R3A, R3.R3B}\ \} \rightarrow \{\ \texttt{R1.R1A, R1.R2B}\ \}$
- $\texttt{R4(}\underline{\texttt{R4A}}\texttt{, R4B)}$

where the primary keys are underlined and the foreign keys denoted by arrows.

1. Map the following SQL query to relational algebra.

```sql
SELECT R2.R2A, R1.R1C
FROM R1, R2, R3
WHERE R3.R3D = "test"
   AND R3.R3A = R1.R1A
   AND R3.R3B = R1.R1B
   AND R3.R3C = R2.R2A
   AND R1.R1D < R3.R3D
```

2. Apply heuristic **logical** query optimization and find an equivalent expression in relational algebra that is efficient in its execution – among other aspects, consider both selection and projection for optimization.
   You can directly state the final result, i.e., there is no need to list the partial results after applying each of the steps.
   You may state your answer represented either as an algebra expression or as an operator tree.

7 Multiple Choice (13 points)

Mark all correct answers! There are no points for partially correct answers but there are also no negative points for wrong answers. You might have to mark **multiple** answers per question as correct.

1. Given relations `R (a:int, x:int)` and `S (b:int, x:int)`.
   Relation `R` contains 10 arbitrary rows, relation `S` contains 3 arbitrary rows. There are no additional restrictions.

   ```
   SELECT * FROM R JOIN S USING X;
   ```

   The result of this query contains at least ___ and at most ___ rows.

   - ☐ 0 and 30
   - ☐ 1 and 30
   - ☐ 3 and 10
   - ☐ 3 and 30

2. For which definitions of table `Person` could the result of the following SQL query contain `NULL` values?

   ```
   SELECT PID
   FROM Person;
   ```

   - ☐ `CREATE TABLE Person(PID INTEGER PRIMARY KEY);`
   - ☐ `CREATE TABLE Person(PID INTEGER PRIMARY KEY DEFAULT 42);`
   - ☐ `CREATE TABLE Person(PID INTEGER PRIMARY KEY NOT NULL);`
   - ☐ `CREATE TABLE Person(PID INTEGER);`

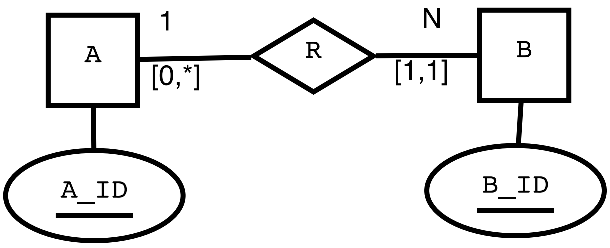
3. Which of the following are benefits of normalizing a relational design?

   - ☐ Modification anomalies are avoided.
   - ☐ The information redundancy in the schema is minimized.
   - ☐ The number of relations in the database schema is minimized.
   - ☐ The number of joins in queries is reduced.

4. Which of the following statements about transactions are true?

- ☐ Consistency means that changes of successfully committed transactions must not be lost.
- ☐ Atomicity means that each transactions is ended with a commit.
- ☐ The WAL principle has a greater influence on durability than on isolation.
- ☐ During log-based recovery we first do *undo* then *redo*.

5. Which combination of the following `CREATE TABLE` statements with integrity constraints corresponds *exactly* to the following ER diagram?



- ☐ `CREATE TABLE A (A_ID INTEGER PRIMARY KEY)`
- ☐ `CREATE TABLE A (A_ID INTEGER PRIMARY KEY, B_ID INTEGER UNIQUE)`
- ☐ `CREATE TABLE B (B_ID INTEGER PRIMARY KEY, A_ID INTEGER NOT NULL, FOREIGN KEY A_ID REFERENCES A(A_ID))`
- ☐ `CREATE TABLE B (B_ID INTEGER PRIMARY KEY, A_ID INTEGER REFERENCES A(A_ID))`

6. Which of the following are valid combinations of clustered, non-clustered, dense, and sparse characteristics of indexes?

- ☐ clustered and dense
- ☐ non-clustered and sparse
- ☐ clustered and sparse
- ☐ non-clustered and dense

7. Natural Joins are derived relational operators that use the following fundamental operators?

   - ☐ rename
   - ☐ selection
   - ☐ difference
   - ☐ projection
   - ☐ union
   - ☐ Cartesian product

8. In a schedule a transaction T_1 writes the value of a data element X. Later the value of X is read by a transaction T_2. To be a *cascadeless* schedule

   - ☐ T_1 must output the log record with the updated value of X to stable storage before T_2 can read X
   - ☐ T_1 must commit before T_2 commits
   - ☐ T_2 must commit before T_1 commits
   - ☐ T_1 must commit before T_2 can read X

9. In a schedule a transaction T_1 writes the value of a data element X. Later the value of X is read by a transaction T_2. To be a *recoverable* schedule

   - ☐ T_1 must output the log record with the updated value of X to stable storage before T_2 can read X
   - ☐ T_1 must commit before T_2 commits
   - ☐ T_2 must commit before T_1 commits
   - ☐ T_1 must commit before T_2 can read X

10. A user executes the following SQL statements in a single transaction on the *empty* table `emp`:

```
BEGIN;
INSERT INTO emp VALUES(1,'Joe');
INSERT INTO emp VALUES(2,'Jim');
INSERT INTO emp VALUES(3,'Jill');
ROLLBACK;
```

   How many tuples are contained in the table afterwards?

   - ☐ 0
   - ☐ 1
   - ☐ 2
   - ☐ 3

11. Does the first column in a create table statement have to be the primary key?

    ☐ true
    ☐ false

12. When using 2-Phase-Locking, what happens if a transaction T_i requests a lock on a data item D that is already locked by another transaction T_k in a non-compatible mode, i.e., T_k holds an exclusive lock on D?

    ☐ T_i and T_k are in a deadlock and nothing will happen
    ☐ T_i is committed
    ☐ T_i continues with the next operation in line and returns to the operation later
    ☐ T_i is delayed
    ☐ T_k is aborted (rolled back)
    ☐ T_k is delayed

13. Is it possible to update more than one attribute in a row with a single SQL update statement?

    ☐ true
    ☐ false