SWE3003 Introduction to Database Systems - Midterm Spring 2024

Student ID	Name

For Instructor/TA only,

Q1	Q2	Q3	Q4	Q5	Q6	Q7	Total

Academic Honor Pledge

I affirm that I will **not** at any time be involved with **cheating** or **plagiarism** while enrolled as a student of **Introduction to Database Systems** class at Sungkyunkwan University. I understand that violation of this code will result in penalties as severe as indefinite suspension from the university.

Your signature: ______________________

1. [30 pts] For each of the following statements, indicate whether it is TRUE or FALSE. You will get 3 points for each correct answer, -3 points for each incorrect answer, and 0 points for each blank answer or both marked answers.

			T	F
(a)	A attribute is a property that describes the relationship between data and the entities that use it.	F	☐	☐
(b)	A candidate key is also a superkey, but there may be superkeys that are not candidate keys.	T	☐	☐
(c)	A weak entity set and its strong entity set must be stored as a single table.	F	☐	☐
(d)	One-to-one relationship set cannot be stored as a single table.	F	☐	☐
(e)	Data dictionary manages statistics about table accesses, such as the number of times a table is accessed, the frequency of particular queries, and other usage patterns.	T	☐	☐
(f)	In the WHERE clause of a SQL query, the condition NULL = NULL is evaluated to be true.	F	☐	☐
(g)	Records cannot be updated in the underlying table through a view.	F	☐	☐
(h)	Using 'Statement' objects helps prevent SQL injection attacks.	F	☐	☐
(i)	You can only create one primary index (clustered index) on a table.	T R	☐	☐
(j)	Secondary indexes (unclustered indexes) are not effective when executing range queries	F	☐	☐

2. [20 pts] Consider the following schema for a medical appointment scheduling program.

   - Doctors(doctorid, dr_name)
   - Patients(patientid, name)
   - Appointments(doctorid, patientid, date)

   Write a relational algebra for the following queries.

   - (a) Find the names of patients who have appointments with 'Dr. Kim' on '2024-04-22':

     answer:

     $$\Pi_{\text{name}}\,(\text{Patients} \bowtie (\sigma_{\text{date}='2024-04-22'}\,(\text{Appointments}) \bowtie \sigma_{\text{dr\_name}='Dr.Kim'}\,(\text{Doctors})))$$

   - (b) Find the names of doctors who have appointments with other doctors on '2024-04-22'. Assume everyone's name is unique.

     answer:

     $$\Pi_{\text{dr\_name}}\,((\sigma_{\text{date}='2024-04-22'}\,(\text{Appointments}) \bowtie \text{Patients}) \bowtie_{\text{name = dr\_name}} \text{Doctors})$$

   - (c) Find the names of doctors who do not have appointments on '2024-04-22'.

     answer:

     $$\Pi_{\text{dr\_name}}\,(\text{Doctors}) - \Pi_{\text{dr\_name}}\,(\text{Doctors} \bowtie \sigma_{\text{date}='2024-04-22'}\,(\text{Appointments}))$$

   - (d) Find the names of patients who have appointments with all doctors.

     answer:

     $$\Pi_{\text{name}}\,((\text{Patients} \bowtie \text{Appointments}) \div \Pi_{\text{doctorid}}\,(\text{Doctors}))$$

     Or

     $$\Pi_{name}\,(Patients \bowtie Appointments) -$$
     $$\Pi_{name}(\Pi_{name}\,(Patients \bowtie Appointments) \times \Pi_{doctorid}\,(Doctors) -$$
     $$(\Pi_{name,doctorid}\,(Patients \bowtie Appointments)))$$

3. [20 pts] Write each of the following queries in SQL for the given relations.

- Doctors(doctorid, dr_name)
- Patients(patientid, name)
- Appointments(doctorid, patientid, date)

(a) Find the dates on which Doctor ID 1 and Patient ID 1 have appointments but Doctor ID 2 and Patient ID 2 do not have appointments.

```
SELECT date
FROM Appointments
WHERE doctorid='1' AND patientid='1'
  AND date NOT IN (SELECT date
                    FROM Appointments
                   WHERE doctorid='2' AND patientid='2')
```

.

(b) Find the names of patients who have appointments with doctors on days when the doctors have 20 or more appointments.

```
With BusyDoctors(doctorid, date) AS
    (SELECT doctorid, date
     FROM Appointments
     GROUP BY doctorid, date
     HAVING count(*) >= 20
    )
SELECT name
FROM Patients NATURAL JOIN Appointments
              NATURAL JOIN BusyDoctors
```

.

(Cont'd) Write each of the following queries in SQL for the given relations.

- Doctors(<u>doctorid</u>, dr_name)
- Patients(<u>patientid</u>, name)
- Appointments(<u>doctorid</u>, <u>patientid</u>, date)

(c) Find the names of patients who have not had any appointments with doctors before '2024-04-22'.

```
SELECT name
FROM Patients
WHERE name NOT IN (
      SELECT name
        FROM Patients NATURAL JOIN Appointments
       WHERE date < '2024-04-22'
);
```

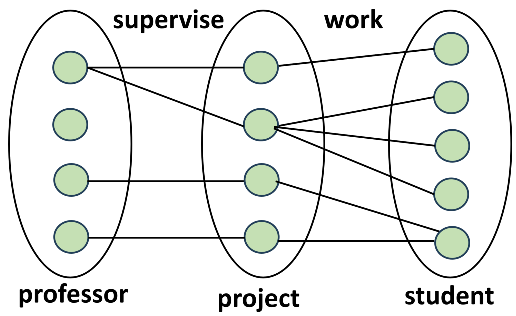
.

(d) Find the names of patients who have appointments with all doctors.

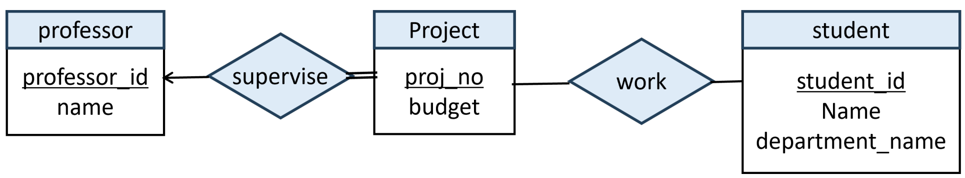
```
SELECT name
  FROM Patients AS P
  WHERE NOT EXISTS (
    (SELECT doctorid
      FROM Doctors)
    EXCEPT
    (SELECT A.doctorid
       FROM Appointments AS A
      WHERE A.patientid = P.patientid
    )
);
```

.

4. [15 pts] The figure below illustrates the relationships between the following entity sets and constraints about cardinality and total/partial participation.
`professor` entity set has <u>`professor_id`</u> and `name` attributes.
`project` entity set has <u>`proj_no`</u>, and `budget` attributes.
`student` entity set has <u>`student_id`</u>, `name`, and `department_name` attributes.



(a) Draw an ER diagram for the three entity sets and two relationship sets.



(b) How many database tables will be needed at minimum? Justify your answer

'supervise' relationship set can be merged into the 'project' entity set since it is one-to-many relationship set. 'work' relationship set can not be merged since it is many-to-many relationship set. So, the answer is 4.

(c) Write a SQL DDL statement to create the '`project`' table. All attributes are of VARCHAR(10) type.

```
CREATE TABLE project (
    professor_id VARCHAR(10)  ,
    proj_no VARCHAR(10),
    budget VARCHAR(10)
    PRIMARY KEY (professor_id, proj_no)
    FOREIGN KEY (professor_id) REFERENCES professor(professor_id)
);
```

partial credits give for the following answer

```
CREATE TABLE project (
    proj_no VARCHAR(10) PRIMARY KEY,
    budget VARCHAR(10)
);
```

5. [15 pts] Consider the following relation and the functional dependency set:

- R(A, B, C, D, E, F, G, H)
- FD = {BC $\rightarrow$ GH, AD $\rightarrow$ E, A $\rightarrow$ H, E $\rightarrow$ BCF, G $\rightarrow$ H}

(a) Find a candidate key of R.

answer: AD

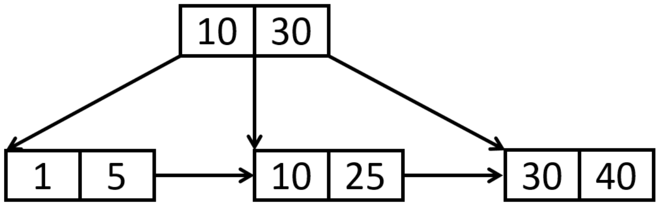
```
AD
-> ADE
-> ADEH
-> ABCDEF
-> ABCDEFGH
```

(b) Decompose the relation R into BCNF.

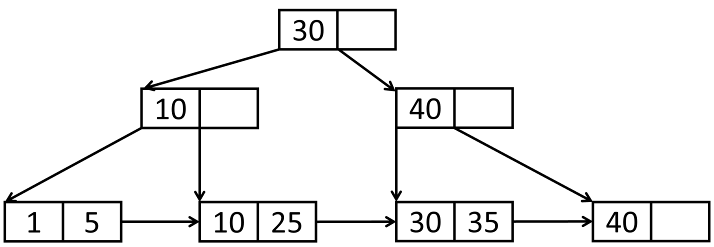
answer: ADE, BCEF, GH, BCG

(c) Is the decomposition lossless? Justify your answer.
answer:
This is a lossless decomposition. It seems we lost BC$\rightarrow$H and A$\rightarrow$H.
But, BC $\rightarrow$ G and G $\rightarrow$ H.
So, we can reconstruct the original tuples when we join GH and BCG. Similarly, A$\rightarrow$H can be reconstructed by joining decomposed tables.

6. [10 pts] Consider the following B+tree with degree 3, i.e, the maximum number of child nodes is 3.



(a) Draw a tree structure after inserting 35 into the tree.



(b) Suppose a single B+ tree node occupies 64 bytes. Each key is 8 bytes, and each pointer is 8 bytes as well. In a B+ tree containing 8 keys, what is the minimum and maximum number of nodes a query might access to find a specific key? Show how you calculated the answer.

Minimum: 2
Maximum: 3

Leaf nodes can have maximum 3 keys.
Internal nodes can have maximum 3 keys and 4 child pointers.
Due to 50% utilization, each leaf must have 2 keys at minimum and each internal node must have one key and 2 child pointers.

So, in the best case, there will be only two levels.

```
                8 key 8  key 8 (key 8) (8)
16 16 16 (8) 8      16 16 16 (8) 8      16 16 16 (8) 8
```

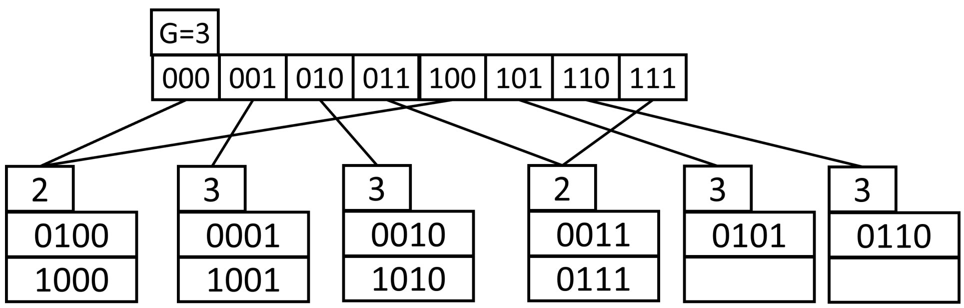
In the worst case, there will be three levels.

```
                                8 key 8  (key 8 key 8) (8)
       8 key 8  (key 8 key 8) (8)                 8 key 8  (key 8 key 8) (8)
16 16 (16) (8) 8      16 16 (16) (8) 8      16 16 (16) (8) 8    16 16 (16) (8) 8
```

7. [10 pts] Suppose each bucket can hold up to 2 records, and the least significant bits are used as the hash key. Suppose you insert the following 10 keys.

`0001, 0010, 0011, 0100, 0101, 0110, 0111, 1000, 1001, 1010`

(1) Draw an extendable hash table after inserting the 10 keys.



(2) Draw a linear hash table after inserting the 10 keys.