02170 Database Systems - Written Exam Part 2, May 2021

This multiple choice constitutes Part 2 of the written examination. Part 1 should be accessed separately.
Both parts have questions concerning the same database (a *Hospital* database). The two parts can be solved in any order, but the *Hospital* database is only described on page 2 in the Part 1 document. So you may need to read that page in order to answer the multiple choice questions.

The multiple choice is based on a "One best answer" concept :

There is always only one correct answer – an answer that is more correct than the others.
Students are only able to select one answer per question.
Every incorrect answer gives 0 points (incorrect answers do not result in subtraction of points).

This question group has no title yet

Conceptual Design

The conceptual design for the *Hospital* database was specified by an Entity-Relationship (E-R) diagram before the relational design (with the relation schemas shown on page 2 in Part 1 of this exam) was made.

We are now going to consider a part of this Entity-Relationship diagram. In order to answer the questions below, you may need to study the text and the tables from page 2 in Part 1 of this exam. For convenience, relevant pieces of that text and relation instances are repeated here:

Each room has a unique *roomNo* and a *capacity* stating the maximal number of patients that can be allocated to (a bed in) that room. Each *patient* has a unique *patientId* and a *patientName*. A *patient* may (or may in a very rare case not) have been allocated to a specific *roomNo* among the registered rooms (to get a bed in that room).

patient(patientId, patientName, roomNo)

patientId	patientName	roomNo
p1	Peter Lund	11
p2	Helen Brown	12
p3	Mary Jones	12
p4	Simon Johnson	NULL

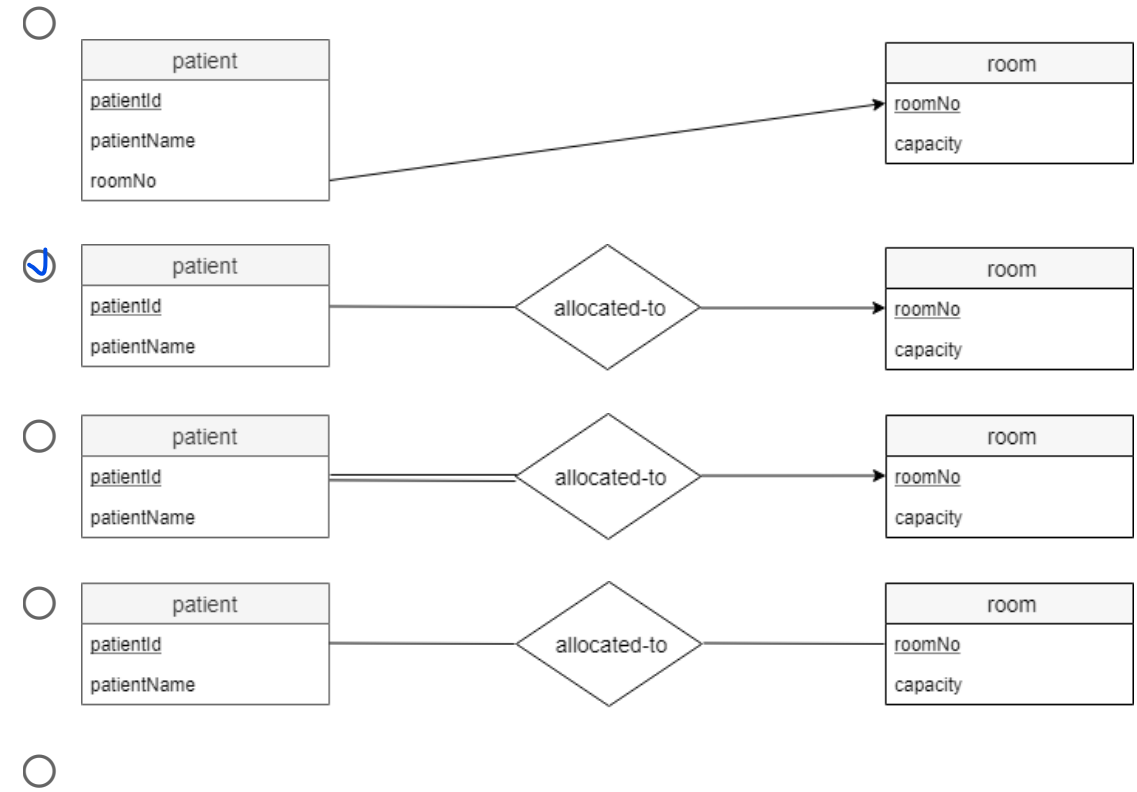
room(roomNo, capacity)

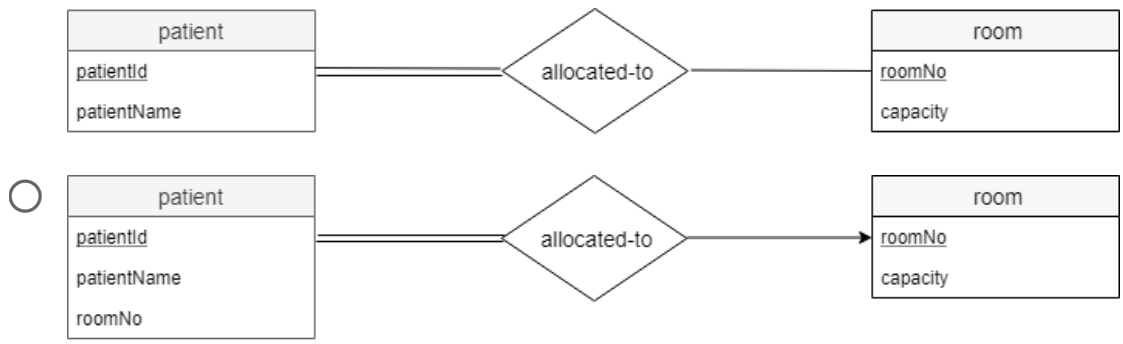
roomNo	capacity
11	2
12	2
13	3

Question 1.1

Which of the following (sub-)diagrams is a correct E-R diagram specifying the *patient* and *room* entity sets and the relationship set *allocated-to* between them?

Choose one answer



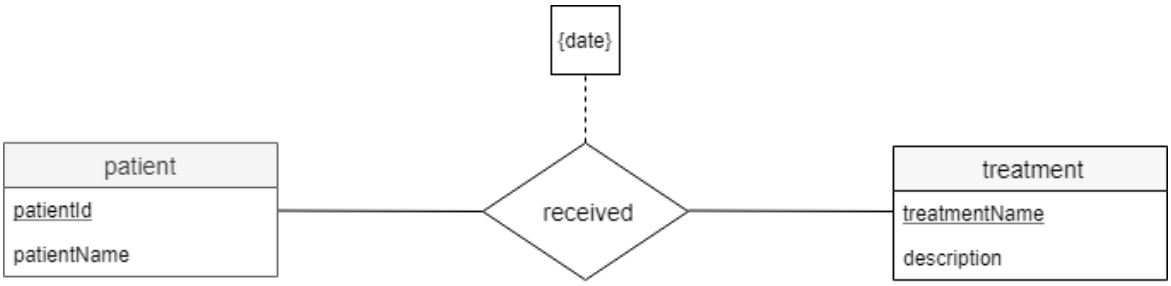

patient
patientId
patientName
allocated-to
room
roomNo
capacity

patient
patientId
patientName
roomNo
allocated-to
room
roomNo
capacity

This question group has no title yet

Entity-Relationship Diagrams and their Conversion to Relation Schemas

At the hospital *patients* can receive *treatments*. The database engineer is going to develop a database to keep track of which treatments the patients have received on which dates. The Entity-Relationship Diagram below is a conceptual model describing entity sets *patient* and *treatment* and a relationship set *received* between them:



Question 2.1

The questions below are concerned with the implications of the depicted cardinality of the *received* relation and the participation of *patient* and *treatment* in *received.*

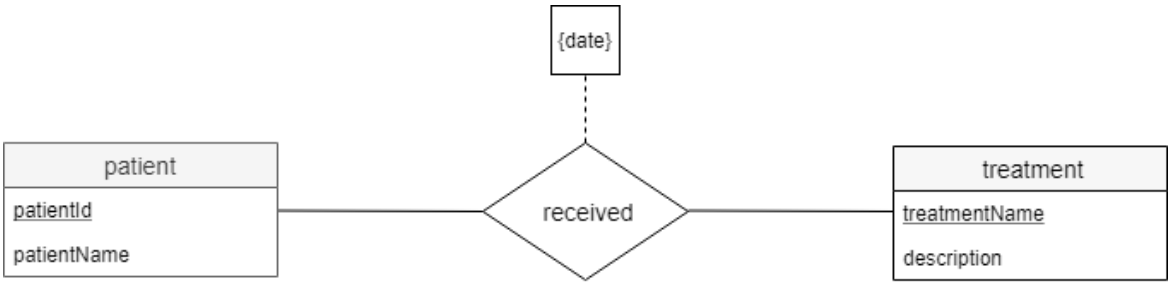
Select the correct answers

	yes	no
Can a patient have received no treatments?	☑	☐
Can a patient have received the same treatment two times on the same date?	☐	☑
Can a patient have received the same treatment on different dates?	☑	☐
Can a patient have received two different treatments on the same date?	☑	☐
Can two different patients have received the same treatment on the same date?	☑	☐

This question group has no title yet

Entity-Relationship Diagrams and their Conversion to Relation Schemas

At the hospital *patients* can receive *treatments*. The database engineer is going to develop a database to keep track of which treatments the patients have received on which dates. The Entity-Relationship Diagram below is a conceptual model describing entity sets *patient* and *treatment* and a relationship set *received* between them:



Question 2.2

Convert the Entity-Relationship diagram shown above into relation schemas using the method described in the textbook and slides adopted in this course. What will the result be (here shown without foreign key constraints)?

Choose one answer

- ○ patient(patientId, patientName), treatment(treatmentName, description), received(patientId, treatmentName, date)
- ✓ patient(patientId, patientName), treatment(treatmentName, description), received(patientId, treatmentName, date)

This question group has no title yet

Normalization

Consider a relation with schema *R(A, B, C, D, E)*, where the primary key is not shown.

The following functional dependencies are assumed:

- A, B → C
- B → D
- C → E

Question 3.1

Which of the following attribute sets are candidate keys for *R*?

Choose one answer

- ☑ There is one candidate key: {A, B}
- ○ There are three candidate keys: {A, B} , {B} and {C}
- ○ There is one candidate key: {A, B, C}

This question group has no title yet

Normalization

Consider a relation with schema *R(A, B, C, D, E)*, where the primary key is not shown.

The following functional dependencies are assumed:

- A, B → C
- B → D
- C → E

Question 3.2

What is the highest normal form of the relation *R* with the schema shown above?

Choose one answer

- [x] 1NF
- [ ] 2NF
- [ ] 3NF
- [ ] 4NF

This question group has no title yet

Normalization

Consider a relation with schema *R(A, B, C, D, E)*, where the primary key is not shown.

The following functional dependencies are assumed:

- A, B → C
- B → D
- C → E

Question 3.3

Choose a primary key for the schema *R* and perform normalization of *R* to at least 4NF (using the method described in the textbook adopted in this course). What is the result of that (ignoring foreign key constraints)?

Choose one answer

- R1(B, D), R2(C, E), R3(A, B, C)
- R1(B, D), R2(A, B, C, E)
- R1(C, E), R2(A, B, C)
- R(A, B, C, D, E)

This question group has no title yet

Integrity Constraints

Question 4.1

Consider the following SQL declaration of the *patient* relation:

```
create table patient (
  patientId varchar(5) primary key,
  patientName varchar(30) not null,
  roomNo smallint,
 foreign key (roomNo) references room(roomNo)
);
```

The following questions concern allowed attribute values.

Select the correct answers

	yes	no
Is *patientId* allowed to be null?	○	☑
Is *roomNo* allowed to be null?	☑	○

This question group has no title yet

Integrity Constraints

Question 4.2

Consider again the following SQL declaration of the *patient* relation:

```
create table patient (
  patientId varchar(5) primary key,
  patientName varchar(30) not null,
  roomNo smallint,
 foreign key (roomNo) references room(roomNo)
);
```

and assume the following content of the *room* and *patient* tables:

room:

roomNo	capacity
11	2
12	2
13	3

patient:

patientId	patientName	roomNo
p1	Peter Lund	11
p2	Helen Brown	12
p3	Mary Jones	12
p4	Simon Johnson	11

What would the results of executing the following SQL statement be: **delete from** *room* **where** *roomNo = 12;*

Choose one answer

- ☑ No rows will be deleted from *room*. No rows will be deleted from *patient.*
- ○ The row having *12* as *roomNo* will be deleted from *room.* No rows will be deleted from *patient.*
- ○ The rows having *12* as *roomNo* will be deleted from *patient.* No rows will be deleted from *room*.
- ○ The row having *12* as *roomNo* will be deleted from *room* and the rows having *12* as *roomNo* will be deleted from *patient.*

This question group has no title yet

Integrity Constraints

Question 4.3

Consider again the following SQL declaration of the *patient* relation:

```
create table patient (
  patientId varchar(5) primary key,
  patientName varchar(30) not null,
  roomNo smallint,
 foreign key (roomNo) references room(roomNo)
);
```

If you should add a referential action to the foreign key constraint, which of the following would be the most natural one:

Choose one answer

- ○ on delete cascade
- ☑ on delete set null

This question group has no title yet

Integrity Constraints

Question 4.4

Consider the following SQL declaration of the *treats* relation which describes which doctors treat which patients, and an example of an instance of that.

```
create table treats(
  doctorId varchar(5),
  patientId varchar(5),
  primary key(doctorId, patientId),
  foreign key(doctorId) references doctor(doctorId),
  foreign key(patientId) references patient(patientId)
);
```

doctorId	patientId
d1	p1
d1	p3
d4	p3

If you should add a referential action to the last foreign key constraint shown above, which of the following would be the most natural one:

Choose one answer

- [x] on delete cascade
- [ ] on delete set null

This question group has no title yet

Formal Relational Query Languages

The following questions consider the relation schemas *room(roomNo, capacity)* and *patient(patientId, patientName, roomNo)* from the *Hospital* database.
Here are examples of relation instances of *room* and *patient*, respectively:

roomNo	capacity
11	2
12	2
13	3

patientId	patientName	roomNo
p1	Peter Lund	11
p2	Helen Brown	12
p3	Mary Jones	12
p4	Simon Johnson	NULL

Question 5.1

Relational Algebra: Which of the following is a correct Relational Algebra expression finding the *roomNo* of those rooms in *room* to which no patients have been allocated according to the information in *patient*?

Choose one answer

- ☑ $\Pi_{roomNo}(room) - \Pi_{roomNo}(patient)$
- ○ $\Pi_{roomNo}(room) \cap \Pi_{roomNo}(patient)$
- ○ $\Pi_{roomNo}(room - patient)$
- ○ $\Pi_{roomNo}(room) \wedge \neg\Pi_{roomNo}(patient)$

This question group has no title yet

Formal Relational Query Languages

The following questions consider the relation schemas *room(roomNo, capacity)* and *patient(patientId, patientName, roomNo)* from the *Hospital* database.
Here are examples of relation instances of *room* and *patient*, respectively:

roomNo	capacity
11	2
12	2
13	3

patientId	patientName	roomNo
p1	Peter Lund	11
p2	Helen Brown	12
p3	Mary Jones	12
p4	Simon Johnson	NULL

Question 5.2

Domain Calculus: Which of the following is a correct Domain Calculus expression finding the *roomNo* of those rooms in *room* to which no patients have been allocated according to the information in *patient*?

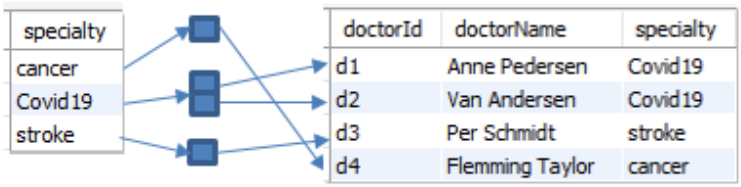
Choose one answer

- $\{< r > \mid \exists\, c\, (< r, c > \in room\ \wedge\ (\exists\, pid, n\, (< pid, n, r > \notin patient)))\}$
- $\{< r > \mid \exists\, c\, (< r, c > \in room\ \wedge\ \neg(\exists\, pid, n\, (< pid, n, r > \in patient)))\}$
- $\{< r > \mid \exists\, c\, (< r, c > \in room\ \wedge\ \neg(\exists\, pid, n, r\, (< pid, n, r > \in patient)))\}$
- $\{< r > \mid \exists\, r, c, pid, n\, (< r, c > \in room\ \wedge\ \neg(< pid, n, r > \in patient))\}$

This question group has no title yet

Indexing and Hashing

When answering the questions on this page, you should use the terminology and theory of the text book and slides adopted in this course.
In these questions it is assumed that the database system does not automatically choose any indices.
In the first two questions, you will be asked about the type of the following index for the *doctor(doctorId, doctorName, specialty)* relation with search key *specialty*:



Question 6.1

Is the index primary or secondary?

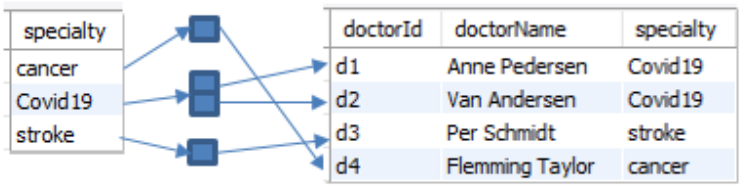
Choose one answer

- ○ primary
- ◉ secondary

This question group has no title yet

Indexing and Hashing

When answering the questions on this page, you should use the terminology and theory of the text book and slides adopted in this course.
In these questions it is assumed that the database system does not automatically choose any indices.
In the first two questions, you will be asked about the type of the following index for the *doctor(doctorId, doctorName, specialty)* relation with search key *specialty*:



Question 6.2

Is the index dense or sparse?

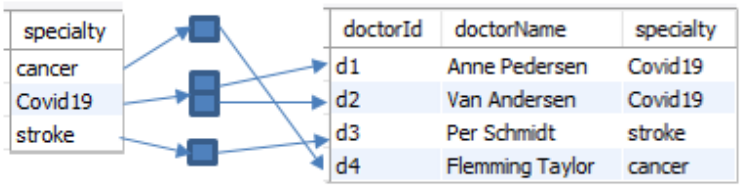
Choose one answer

- [x] dense
- [ ] sparse

This question group has no title yet

Indexing and Hashing

When answering the questions on this page, you should use the terminology and theory of the text book and slides adopted in this course.
In these questions it is assumed that the database system does not automatically choose any indices.
In the first two questions, you will be asked about the type of the following index for the *doctor(doctorId, doctorName, specialty)* relation with search key *specialty*:



Question 6.3

Assume that the database management system does not automatically choose indices, and that the database statistics show that a very common query is of the form:

SELECT roomNo **FROM** patient **WHERE** patientName = *n*,

where *n* is some name.

Under these assumptions, which of the following would you do to make these queries faster?

Choose one answer

- ○ I would use hashing with search key *roomNo*.
- ○ I would make an index on *roomNo.*
- ✓ I would use hashing with search key *patientName*.
- ○ I would make an index on *patientName*.