CS4221/CS5421

Test 1

INSTRUCTIONS

1. This assessment **starts at 18:40**.
2. This assessment **ends at 20:25**.
3. The **maximum mark is 50 for 25% of the final mark.**
4. This is an **open book, open computer, but closed Internet assessment**.
5. You are not allowed to communicate with anyone but members of the teaching team. Shall you use any external source of information make sure that you include a reference to the source in your answer (e.g. the URL).
6. Any student who is alleged to have committed or attempted to commit, or caused or attempted to cause any other person to commit any of the following offences: plagiarism, giving or receiving unauthorised assistance in academic work, or other forms of academic dishonesty, may be subject to disciplinary proceedings.
7. Within the 10 minutes before the start of the assessment, **download this question paper and the template answer file**:
   - "`test1.pdf`",
   - "`answer.txt`",

   from the Luminus directory:

   "`Files > Tests > Test 1`".
8. Within the 10 minutes before the start of the assessment, **download the following files**:
   - "`code.sql`",

   from the Luminus directory:

   "`Files > Tests > Test 1 > Code`".
9. **Write your student number** in the corresponding section of the file "`answer.txt`".
10. **Write your answers** to the questions in the corresponding sections of the file "`answer.txt`".
11. **Within the 5 minutes following the end of the assessment, upload the files:**
    - "`answer.txt`",
    - "`query.svg`",

    to the Luminus directory:

    "`Files > Tests > Test 1 > Submissions`".

Question 1 (5 points)

Consider the entity-relationship diagram of Figure 1.

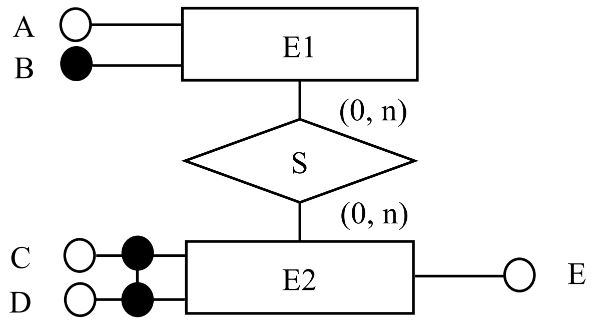

Figure 1: Entity-relationship diagram.

(a) (2 points) Find the set Σ of functional dependencies on the relation $R = \{A, B, C, D, E\}$ captured by the entity-relationship diagram. Σ should be compact and minimal. Do not show the steps.

> **Solution:**
> $$\Sigma = \{\{B\} \rightarrow \{A\}, \{C, D\} \rightarrow \{E\}\}$$

(b) (3 points) Propose a lossless and dependency preserving decomposition in Boyce-Codd normal form of R with the dependencies captured by the entity-relationship diagram. Indicate the candidate keys of each fragment. Do not show the steps.

> **Solution:** $R_1 = \{A, \underline{B}\}$, $R_2 = \{\underline{C, D}, E\}, R_3 = \{\underline{B, C, D}\}$

Question 2 (12 points)

Consider the entity-relationship diagram of Figure 2.

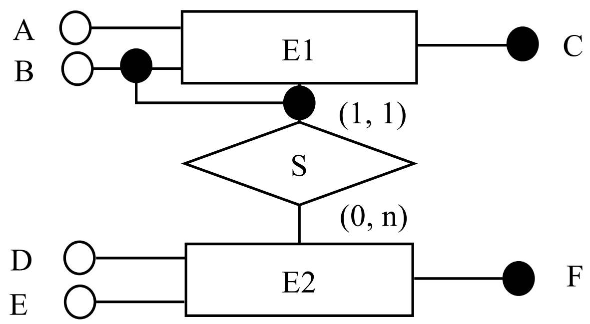

Figure 2: Entity-relationship diagram.

(a) (5 points) Find two different but equivalent sets Σ_1 and Σ_2 of functional dependencies on the relation $R = \{A, B, C, D, E, F\}$ captured by the entity-relationship diagram. Σ_1 and Σ_2 should be compact and minimal.

> **Solution:**
> $$\Sigma_1 = \{\{C\} \rightarrow \{A, B, F\}, \{B, F\} \rightarrow \{C\}, \{F\} \rightarrow \{D, E\}\}$$
> $$\Sigma_2 = \{\{B, F\} \rightarrow \{A, C\}, \{C\} \rightarrow \{B, F\}, \{F\} \rightarrow \{D, E\}\}$$

(b) (3 points) Propose a lossless and dependency preserving decomposition in Boyce-Codd normal form of R with the dependencies captured by the entity-relationship diagram. You do not need to indicate the candidate keys of the fragments. Do not show the steps.

> **Solution:** $R_1 = \{A, \underline{C}, \underline{B, F}\}$, $R_2 = \{D, E, \underline{F}\}$

(c) (4 points) Translate the entity-relationship diagram into the corresponding SQL DDL statements following the translation process outlined in the lecture. Use the domain `INT` for all the attributes. Enforce all the constraints to your knowledge.

> **Solution:**

```
 CREATE TABLE R12 (
 D INT NOT NULL,
 E INT NOT NULL,
 F INT PRIMARY KEY);

CREATE TABLE R11(
A INT NOT NULL,
B INT NOT NULL,
C INT PRIMARY KEY,
F INT NOT NULL REFERENCES R12(F),
UNIQUE (B, F));
```

or

```
CREATE TABLE R22 (
D INT NOT NULL,
E INT NOT NULL,
F INT PRIMARY KEY);

CREATE TABLE R21 (
A INT NOT NULL,
B INT NOT NULL,
C INT UNIQUE NOT NULL,
F INT NOT NULL REFERENCES R22(F),
PRIMARY KEY (B, F));
```

Question 3 (13 points)

Consider the relation $R = \{A, B, C, D, E\}$ with the set of dependencies $\Sigma = \{\{A, C\} \rightarrow \{B, D, E\}, \{B\} \rightarrow \{A\}\}$. We decompose R into the three fragments $R_1 = \{A, B, C\}$, $R_2 = \{B, C, D\}$, and $R_3 = \{A, C, E\}$.

(a) (2 points) Find a set compact and minimal set Σ_1 of the projected dependencies on R_1. Do not show the steps.

> **Solution:** $\Sigma_1 = \{\{A, C\} \rightarrow \{B\}, \{B\} \rightarrow \{A\}\}$

(b) (1 point) Find a set compact and minimal setΣ_2 of projected dependencies on R_2. Do not show the steps.

> **Solution:** $\Sigma_1 = \{\{B, C\} \rightarrow \{D\}\}$

(c) (1 point) Find a set compact and minimal set Σ_3 of projected dependencies on R_3. Do not show the steps.

Solution: $\Sigma_2 = \{\{A, C\} \rightarrow \{E\}\}$

(d) (1 point) Is the decomposition of R into the three fragments R_1, R_2, and R_3 dependency preserving? Answer "yes" or "no". If you answer "yes", do not give any explanation. If you answer "no", indicate one functional dependency lost in the decomposition.

Solution: Yes.

(e) (2 points) Use the Chase algorithm with distinguished variables to test whether the decomposition of R into the three fragments R_1, R_2, and R_3 is lossless for Σ. Show the steps.

Solution: We set the chase for checking whether the decomposition is lossless.

```
A B C D E
a a a _ _
_ a a a _
a _ a _ a
```

We use $\{B\} \rightarrow \{A\}$.

```
A B C D E
a a a _ _
a a a a _
a _ a _ a
```

We use $\{A, C\} \rightarrow \{B, D, E\}$.

```
A B C D E
a a a a a
a a a a a
a a a a a
```

The decomposition is lossless.

or

We set the chase for checking whether the decomposition is lossless.

```
A B C D E
a a a _ _
_ a a a _
a _ a _ a
```

We use $\{A, C\} \rightarrow \{B, D, E\}$.

```
A B C D E
a a a _ a
_ a a a _
a a a _ a
```

We use $\{B\} \rightarrow \{A\}$.

```
A B C D E
a a a _ _
a a a a _
a a a _ a
```

We use $\{A, C\} \rightarrow \{B, D, E\}$.

```
A B C D E
a a a a a
a a a a a
a a a a a
```

The decomposition is lossless.

(f) (2 points) Use the Chase algorithm with distinguished variables to test whether the decomposition of R into the three fragments R_1, R_2, and R_3 is lossless for $\Sigma_1 \cup \Sigma_2 \cup \Sigma_3$. Show the steps.

Solution: We set the chase for checking whether the decomposition is lossless.

```
A B C D E
a a a _ _
_ a a a _
a _ a _ a
```

We use $\{B\} \rightarrow \{A\}$.

```
A B C D E
a a a _ _
a a a a _
a _ a _ a
```

We use $\{A, C\} \rightarrow \{E\}$.

```
A B C D E
a a a _ a
a a a a a
a _ a _ a
```

We use $\{A, C\} \rightarrow \{B\}$.

```
A B C D E
a a a _ a
a a a a a
a a a _ a
```

We use $\{B, C\} \rightarrow \{D\}$.

```
A B C D E
a a a a a
a a a a a
a a a a a
```

The decomposition is lossless.

(g) (4 points) Prove, using the three Armstrong axioms only, that $\Sigma \models \{B, C\} \rightarrow \{A, D, E\}$.

Solution:

1. We know that $\{A, C\} \rightarrow \{B, D, E\}$.
2. We know that $\{B\} \rightarrow \{A\}$.
3. Therefore $\{A, C\} \rightarrow \{A, B, D, E\}$ by Augmentation of (1) with $\{A\}$.
4. We know that $\{A, D, E\} \subset \{A, B, D, E\}$.
5. Therefore $\{A, B, D, E\} \rightarrow \{A, D, E\}$ by Reflexivity with (4).
6. Therefore $\{A, C\} \rightarrow \{A, D, E\}$ by Transitivity of (3) and (5).
7. Therefore $\{B, C\} \rightarrow \{A, C\}$ by Augmentation of (2) with $\{C\}$.
8. Therefore $\{B, C\} \rightarrow \{A, D, E\}$ by Transitivity of (7) and (6).

   Q.E.D.

Question 4 (8 points)

Consider the relation $R = \{A, B, C, D, E\}$ with the set of dependencies $\Sigma = \{\{A, B\} \twoheadrightarrow \{C\}, \{A, B\} \rightarrow \{D\}\}$.

(a) (4 points) Prove, using the three Armstrong axioms and Complementation, Augmentation, Transitivity, Replication and Coalescence for multivalued dependencies only, that $\Sigma \models \{A, B\} \twoheadrightarrow \{C, D\}$
.

Solution:

1. We know that $\{A, B\} \twoheadrightarrow \{C\}$.
2. We know that $\{A, B\} \rightarrow \{D\}$.
3. Therefore that $\{A, BC\} \rightarrow \{C, D\}$ by Augmentation of (2) with $\{C\}$.
4. $\{A, B\} \twoheadrightarrow \{C, D\}$ by Coalescence of (1) and (3).

   Q.E.D.

or

1. We know that $\{A, B\} \twoheadrightarrow \{C\}$.
2. We know that $\{A, B\} \rightarrow \{D\}$.
3. Therefore that $\{A, B\} \twoheadrightarrow \{D\}$ by Replication of (2).
4. Therefore $\{A, B\} \twoheadrightarrow \{A, B, C\}$ by Augmentation of (1) with $\{A, B\}$.
5. Therefore $\{A, B, C\} \twoheadrightarrow \{C, D\}$ by Augmentation of (3) with $\{C\}$.
6. $\{A, B, C\} \twoheadrightarrow \{E\}$ by Complementation of (5).
7. $\{A, B\} \twoheadrightarrow \{E\}$ by Transitivity of (3) and (6).
8. $\{A, B\} \twoheadrightarrow \{C, D\}$ by Complementation of (7).

   Q.E.D.

(b) (4 points) Chase $\{A, B\} \twoheadrightarrow \{C, D\}$ in R with Σ.

Solution: We set the chase for $\{A, B\} \twoheadrightarrow \{C, D\}$

A B C B E
```
1 1 1 1 1
1 1 2 2 2
```
We use $\{A, B\} \rightarrow \{D\}$.
```
A B C D E
1 1 1 1 1
1 1 2 1 2
```
We use $\{\{A, B\} \twoheadrightarrow \{C\}$.
```
A B C D E
1 1 1 1 1
1 1 2 1 2
1 1 2 1 1
1 1 1 1 2
```
We see that $\{A, B\} \twoheadrightarrow \{C, D\}$.

Question 5 (12 points)

We want to create a table `R` with column `A`, `B`, `C`, `D`, `E` and domains `INT`.

The following PLSQL stored function generates a random integer between the provided minimum and maximum.

```
CREATE OR REPLACE FUNCTION random_between(min INT ,max INT)
   RETURNS INT AS
$$
BEGIN
```

```
   RETURN floor(random()* (max-min + 1) + min);
END;
$$ language 'plpgsql';
```

The following query prints 5 random natural numbers between 1 and 10.

```
SELECT random_between(1, 10) AS A
FROM generate_series(1,5);
```

We use the following code to populate R.

```
INSERT INTO R (A, B, C, D, E) SELECT DISTINCT
    random_between(0, 999) AS A,
    random_between(0, 9) AS B,
        0 AS C,
        0 AS D,
    0 AS E
FROM generate_series(1, 10000);

UPDATE R SET C = A + (B * 1000);
UPDATE R SET D = MOD(C, 2);
UPDATE R SET E = MOD(C, 3);
```

Notice that several constraints are imposed by this random generation. For instance, the values of A are between 1 and 99. For instance $\{C\} \rightarrow \{D\}$.

(a) (4 points) Create and populate a table R with column A, B, C, D, E, F with domains INT and with the integrity constraints that can be inferred from the population code. Give the CREATE TABLE code for R only. Do not give the rest of the code populating R (note that you may need an intermediary table).

Solution:

```
CREATE TABLE R (
        A INT CHECK(A BETWEEN 0 AND 999) NOT NULL,
        B INT CHECK(B BETWEEN 0 AND 9) NOT NULL,
        C INT CHECK(C BETWEEN 0 AND 9999) PRIMARY KEY,
        D INT CHECK(D BETWEEN 0 AND 1) NOT NULL,
        E INT CHECK(E BETWEEN 0 AND 2) NOT NULL,
    UNIQUE (A, B));
```

The following code needs to be run in the following order.

```
CREATE OR REPLACE FUNCTION random_between(min INT ,max INT)
   RETURNS INT AS
$$
BEGIN
   RETURN floor(random()* (max-min + 1) + min);
END;
$$ language 'plpgsql';

SELECT random_between(1, 10) AS A
FROM generate_series(1,5);

CREATE TABLE R1 (
        A INT,
        B INT,
        C INT,
        D INT,
        E INT);
```

```
INSERT INTO R1 (A, B, C, D, E) SELECT DISTINCT
   random_between(0, 999) AS A,
   random_between(0, 9) AS B,
      0 AS C,
      0 AS D,
   0 AS E
FROM generate_series(1, 10000);

UPDATE R1 SET C = A + (B * 1000);
UPDATE R1 SET D = MOD(C, 2);
UPDATE R1 SET E = MOD(C, 3);

CREATE TABLE R (
      A INT CHECK(A BETWEEN 0 AND 999) NOT NULL,
      B INT CHECK(B BETWEEN 0 AND 9) NOT NULL,
      C INT CHECK(C BETWEEN 0 AND 9999) PRIMARY KEY,
      D INT CHECK(D BETWEEN 0 AND 1) NOT NULL,
      E INT CHECK(E BETWEEN 0 AND 2) NOT NULL,
   UNIQUE (A, B));

INSERT INTO R (SELECT * FROM R1);
```

(b) (2 points) Write a query on `R` for which PostgreSQL plans a sequential scan on `R` (`Seq Scan`) for the current data. Write the query and copy the plan obtained by `EXPLAIN`.

Solution:

```
EXPLAIN SELECT * FROM R WHERE d = 0;

"Seq Scan on r (cost=0.00..120.09 rows=3199 width=20)"
" Filter: (d = 0)"
```

(c) (2 points) Write a query on R for which PostgreSQL plans to use an index scan on one of `R`'s indexes (`Ind Scan`) for the current data. Write the query and copy the plan obtained by `EXPLAIN`.

Solution:

```
EXPLAIN SELECT * FROM R WHERE c = 0;

"Index Scan using r_pkey on r (cost=0.28..8.30 rows=1 width=20)"
" Index Cond: (c = 0)"

EXPLAIN SELECT * FROM R WHERE c = 0;

"Index Scan using r_pkey on r (cost=0.28..8.30 rows=1 width=20)"
" Index Cond: (c = 0)"
```

or

```
EXPLAIN SELECT * FROM R WHERE a= 0 AND b = 0;

"Index Scan using r_a_b_key on r (cost=0.28..8.30 rows=1 width=20)"
" Index Cond: ((a = 0) AND (b = 0))"
```

or

This only works after Postgres has build some statistics.

```
EXPLAIN SELECT * FROM R WHERE a = 0;

"Index Scan using r_a_b_key on r (cost=0.28..20.67 rows=8 width=20)"
" Index Cond: (a = 0)"
```

(d) (4 points) Can you rewrite the following query into an equivalent query that runs faster?

```
SELECT R0.C
FROM R R0
WHERE R0.E = 0
AND R0.C NOT IN (SELECT R1.C
FROM R R1
WHERE R1.E = R0.E
       AND (R1.C, R1.A) IN (
          SELECT R2.C, R2.C
          FROM R R2
          WHERE R2.A = R2.C
          AND R1.B <> R2.B));
```

Give the new query. Do not comment or explain.

Solution: You can reduce the size of R for testing.

```
DELETE FROM R WHERE random() >.01;
```

You can immediately change `R1.E = R0.E` to `R1.E = 0`.
The original query after the change takes `"Planning Time: 0.342 ms"` and `"Execution Time: 1965.82 ms"`.
The original query can be further simplified by unnesting and under the knowledge of the functional dependencies.

```
SELECT R0.C
FROM R R0
WHERE R0.E = 0;
```

The rewritten query takes `"Planning Time: 0.136 ms"` and `"Execution Time: 4.181 ms"`

– END OF PAPER –