NATIONAL UNIVERSITY OF SINGAPORE

SCHOOL OF COMPUTING
FINAL ASSESSMENT FOR
Semester 2 AY2024/25

IT5008 Database Design and Programming

07 May 2025 Time Allowed 120 minutes

INSTRUCTIONS TO CANDIDATES

1. This assessment paper contains **25** questions and comprises **8** printed pages, including this page.
2. More specific instructions *may* be given at the beginning of each section. Please read them carefully.
3. Write all your answers in the answer sheet provided. Please read the additional instruction for multiple choice (MCQ) and multiple response (MRQ) questions.
   - Any answer not in the space provided will not be graded.
   - For MCQ and MRQ, **shade your answer in the corresponding bubble on the answer sheet**.
   - For MCQ, if multiple answers are equally appropriate, pick one and shade **ONLY** the chosen answer on the answer sheet. Do **NOT** shade more than one answer.
   - For MRQ, shade **ALL** correct answers. Partial marks *may* be given, but not guaranteed.
   - If there are no correct answer or no appropriate choices, shade X in the answer sheet. No partial mark given if X is chosen.
   - Please erase the answers completely before shading new options.
4. Shade and write your student number in the answer sheet. Do **NOT** write your name.
5. The total marks for this assessment is **80 points**.
6. This is a **CLOSED-BOOK** assessment. You are only allowed to refer to one double-sided A4-size paper.
7. All SQL statements in this assessment paper are written for PostgreSQL 16.
8. We will use the monospace `NULL` to represent NULL-values. We will represent string with single-quote `'string'`.
9. The last **1** page(s) is/are left blank. You may use them for your own work.
10. Submit only the answer sheet at the end of the assessment. You may keep the question paper.

Question	Points
1 - 11	30
12 - 21	30
22 - 24	9
25	11
TOTAL	80

Functional Dependencies and Normal Forms

For the next **eleven (11)** questions, we will be using the following relation and set of functional dependencies.

$R = \{A, B, C, D, E\}$

$\Sigma = \{\, \{B, C, D\} \to \{E\}, \{B, E\} \to \{A, B\}, \{B, D, E\} \to \{B, E\}, \{B, D\} \to \{B, C\}, \{E\} \to \{D\} \,\}$

Notes: We start with the following computation:

- **Keys:** $\{\, \{B, D\}, \{B, E\} \,\}$
- **Minimal Cover:** $\{\, \{B, D\} \to \{C\}, \{B, D\} \to \{E\}, \{B, E\} \to \{A\}, \{E\} \to \{D\} \,\}$

1. (2 points) **(MRQ)** Select **ALL** the *trivial* functional dependencies in Σ.

   A. $\{B, C, D\} \to \{E\}$
   B. $\{B, E\} \to \{A, B\}$
   ✓ $\{B, D, E\} \to \{B, E\}$
   D. $\{B, D\} \to \{B, C\}$
   E. $\{E\} \to \{D\}$

   Select X if there is no answer.

   Notes: $\{B, E\} \subseteq \{B, D, E\}$

2. (2 points) **(MRQ)** Select **ALL** the *non-trivial* functional dependencies in Σ that are not *completely non-trivial*.

   A. $\{B, C, D\} \to \{E\}$
   ✓ $\{B, E\} \to \{A, B\}$
   C. $\{B, D, E\} \to \{B, E\}$
   ✓ $\{B, D\} \to \{B, C\}$
   E. $\{E\} \to \{D\}$

   Select X if there is no answer.

   Notes: **B:** $\{A, B\} \not\subseteq \{B, E\} \wedge \{A, B\} \cap \{B, E\} \neq \emptyset$; **D:** $\{B, C\} \not\subseteq \{B, D\} \wedge \{B, C\} \cap \{B, D\} \neq \emptyset$

3. (2 points) **(MRQ)** Select **ALL** the *completely non-trivial* functional dependencies in Σ.

   ✓ $\{B, C, D\} \to \{E\}$
   B. $\{B, E\} \to \{A, B\}$
   C. $\{B, D, E\} \to \{B, E\}$
   D. $\{B, D\} \to \{B, C\}$
   ✓ $\{E\} \to \{D\}$

   Select X if there is no answer.

   Notes: **A:** $\{E\} \cap \{B, C, D\} = \emptyset$; **E:** $\{E\} \cap \{D\} = \emptyset$

4. (3 points) **(MRQ)** Select **ALL** the *superkeys* of R with Σ from the choice below.

   A. $\{A, B, C\}$
   ✓ $\{B, C, D\}$
   C. $\{C, D, E\}$
   D. $\{A, D, E\}$
   ✓ $\{A, B, E\}$

   Select X if there is no answer.

   Notes: **A:** $\{B, C, D\} \supseteq \{B, D\}$; **E:** $\{A, B, E\} \supseteq \{B, E\}$

5. (3 points) **(MRQ)** Select **ALL** the *candidate keys* (i.e., keys) of R with Σ from the choice below.

   A. $\{A, B\}$
   B. $\{B, C\}$
   C. $\{C, D\}$
   D. $\{D, E\}$
   E. $\{A, E\}$

   Select X if there is no answer.

   Notes: Answer is X. There are two keys: $\{B, D\}$ and $\{B, E\}$.

6. (3 points) **(MRQ)** Select **ALL** the *prime attributes* or R with Σ.

   A. A
   ✓ B
   C. C
   ✓ D
   ✓ E

   Select X if there is no answer.

   Notes: $\{B, D\} \cup \{B, E\} = \{B, D, E\}$.

7. (3 points) **(MRQ)** Select **ALL** the functional dependencies below that violates the BCNF property of R with Σ.

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

   Select X if there is no answer.

   Notes: $\{D\} \not\subseteq \{E\}$ and $\{E\}$ is not a candidate key.

8. (3 points) **(MRQ)** Select **ALL** the functional dependencies below that violates the 3NF property of R with Σ.

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

   Select X if there is no answer.

   Notes: Similar to above, but now we cannot use $\{E\} \rightarrow \{D\}$ because D is a prime attribute.

9. (3 points) **(MRQ)** Form one minimal cover of Σ using only the functional dependencies listed below.

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

   Select X if there is no answer.

   Notes: Starting only from Σ, there is only one minimal cover.

10. (4 points) Find one lossless-join decomposition of R with Σ in BCNF using the BCNF decomposition algorithm introduced in the course.

    Show your work. Recap that you need to show that (i) the relation is not in BCNF with respect to its set of functional dependencies and (ii) correctly decompose the relation and compute the functional dependencies. Show both (i) and (ii) for each step of the decomposition.

> **Notes:** Using $\{E\} \rightarrow \{D\}$, we compute $\{E\}^+ = \{D, E\}$. We decompose R into.
>
> - $R_1 = \{D, E\}$ with $\Sigma|_{R_1} = \{ \{E\} \rightarrow \{D\} \}$
> - $R_2 = \{A, B, C, E\}$ with $\Sigma|_{R_2} = \{ \{B, E\} \rightarrow \{A, C\} \}$
>
> Both are in BCNF.

11. (2 points) Is your decomposition in question 10 a dependency-preserving decomposition?

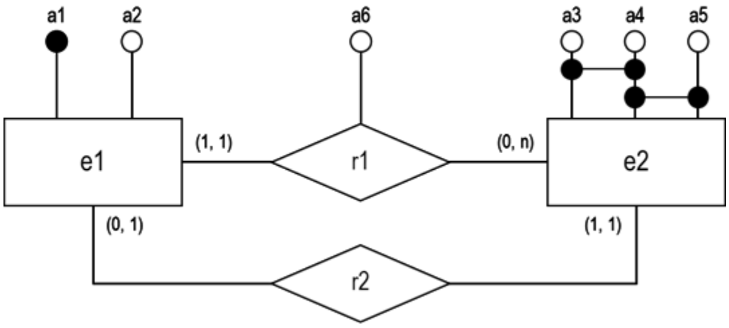
    Show your work. If it is not dependency-preserving, show the functional dependencies from Σ that is not preserved. If it is dependency-preserving, briefly show that all functional dependencies in Σ are preserved by showing the attribute closure of each left-hand side of functional dependencies in Σ.

> **Notes:** It is not dependency-preserving.
>
> From above, we have $\Sigma|_{R_1} = \{ \{E\} \rightarrow \{D\} \}$ and $\Sigma|_{R_2} = \{ \{B, E\} \rightarrow \{A, C\} \}$.
>
> $$\Sigma|_{R_\cup} = \{ \{E\} \rightarrow \{D\} \} \cup \{ \{B, E\} \rightarrow \{A, C\} \} = \{ \{E\} \rightarrow \{D\}, \{B, E\} \rightarrow \{A, C\} \}$$
>
> Consider $\{B, D\} \rightarrow \{C\}$. Since $\{B, D\}^+ = \{B, D\}$ and $\{C\} \not\subseteq \{B, D\}^+$, $\{B, D\} \rightarrow \{C\}$ is not preserved.

Continue on the next page...

Entity-Relationship Diagram

For the next **ten (10)** questions, we will be using the following relation entity-relationship diagram.



A functional dependency holds on the entity-relationship diagram if all the tables and their natural join in all instances of the design do not violate the functional dependency. Note that we do not worry about existence of entries in a relationship in (0, _) cardinality as we assume that there should be no `NULL` values even in the natural join.

Let u and v be entity sets or relationship sets. If during the schema translation, the table for u and the table for v are merged, we refer to it as the table of u or the table of v. Any properties that is true for the table for u should also be true for table for v and vice versa.

Notes: Although we can derive the functional dependencies independently, we can also use the schema to derive them.

```
CREATE TABLE e1r1 (
  a1 INT PRIMARY KEY,
  a2 INT NOT NULL,
  a3 INT NOT NULL,
  a4 INT NOT NULL,
  a6 INT NOT NULL,
  FOREIGN KEY (a3, a4) REFERENCES e2r2(a3, a4) -- need ALTER TABLE
);

CREATE TABLE e2r2 (
  a1 INT NOT NULL UNIQUE REFERENCES e1r1(a1), -- candidate key
  a3 INT,
  a4 INT,
  a5 INT NOT NULL,
  PRIMARY KEY (a3, a4),                       -- primary key
  UNIQUE (a4, a5)                             -- candidate key
);
```

From here, we can derive the following set of functional dependencies.

- $\{a_1\} \rightarrow \{a_2\}$
- $\{a_1\} \rightarrow \{a_3\}$
- $\{a_1\} \rightarrow \{a_4\}$
- $\{a_1\} \rightarrow \{a_5\}$
- $\{a_1\} \rightarrow \{a_6\}$
- $\{a_3, a_4\} \rightarrow \{a_1\}$
- $\{a_3, a_4\} \rightarrow \{a_2\}$
- $\{a_3, a_4\} \rightarrow \{a_5\}$
- $\{a_3, a_4\} \rightarrow \{a_6\}$
- $\{a_4, a_5\} \rightarrow \{a_1\}$
- $\{a_4, a_5\} \rightarrow \{a_2\}$
- $\{a_4, a_5\} \rightarrow \{a_3\}$
- $\{a_4, a_5\} \rightarrow \{a_6\}$

Notice something weird. We can have the following valid instances for `e1r1` and `e2r2`.

`e1r1`

a1	a2	a3	a4	a6
1	_	1	1	_
2	_	1	1	_

`e2r2`

a1	a3	a4	a5
2	1	1	_
1	2	2	_

`e1r1` ⋈ `e2r2`

a1	a2	a3	a4	a5	a6
2	_	1	1	_	_

How can `(a3,a4)` uniquely identify `(a1)` then? The key here is (*as mentioned in the question*) to create a natural join of all tables (on the right) We do enforce the FD but only on the natural join.

12. (3 points) **(MRQ)** Which of the following properties are true in at least one schema translation? Select **ALL** answers that apply.
    - ✓ `(a1)` is a foreign key in the table for `e2`. `(a1)` is referencing `(a1)` in the table for `e1`.
    - B. `(a2)` is a foreign key in the table for `e2`. `(a2)` is referencing `(a2)` in the table for `e1`.
    - ✓ `(a1)` is a foreign key in the table for `e2`. `(a1)` is referencing `(a1)` in the table for `r1`.
    - D. `(a2)` is a foreign key in the table for `e2`. `(a2)` is referencing `(a2)` in the table for `r1`.
    - E. `(a6)` is a foreign key in the table for `e2`. `(a6)` is referencing `(a6)` in the table for `r1`.

    Select X if there is no answer.

    Notes: Because `e1` and `r1` are merged.

13. (3 points) **(MRQ)** Which of the following properties are true in at least one schema translation? Select **ALL** answers that apply.
    - ✓ `(a3,a4)` is a foreign key in the table for `e1`. `(a3,a4)` is referencing `(a3 a4)` in the table for `e2`.
    - ✓ `(a4,a5)` is a foreign key in the table for `e1`. `(a4,a5)` is referencing `(a4,a5)` in the table for `e2`.
    - C. `(a1)` is a foreign key in the table for `e1`. `(a1)` is referencing `(a4, a5)` in the table for `r1`.
    - ✓ `(a3,a4)` is a foreign key in the table for `e1`. `(a3,a4)` is referencing `(a3,a4)` in the table for `r2`.
    - ✓ `(a4,a5)` is a foreign key in the table for `e1`. `(a4,a5)` is referencing `(a4,a5)` in the table for `r2`.

    Select X if there is no answer.

    Notes: Because `e2` and `r2` are merged.

14. (3 points) **(MRQ)** Which of the following properties are true in at least one schema translation? Select **ALL** answers that apply.
    - ✓ The primary key for the table for `r1` is `(a1)`.
    - B. The primary key for the table for `r1` is `(a3,a4)`.
    - C. The primary key for the table for `r1` is `(a4,a5)`.
    - D. The primary key for the table for `r1` is `(a1,a3,a4)`.
    - E. The primary key for the table for `r1` is `(a1,a4,a5)`.

    Select X if there is no answer.

    Notes: Because of $(1, 1)$ cardinality between `e1` and `r1`.

15. (3 points) **(MRQ)** Which of the following properties are true in at least one schema translation? Select **ALL** answers that apply.
    - ✓ The primary key for the table for `r2` is `(a1)`.
    - ✓ The primary key for the table for `r2` is `(a3,a4)`.
    - ✓ The primary key for the table for `r2` is `(a4,a5)`.
    - D. The primary key for the table for `r2` is `(a1,a3,a4)`.
    - E. The primary key for the table for `r2` is `(a1,a4,a5)`.

    Select X if there is no answer.

Notes: Because of (1, 1) cardinality between `e2` and `r2` for **B** and **C**. Because of (0, 1) cardinality between `e1` and `r2` for **A**.

16. (3 points) **(MRQ)** Select **ALL** the functional dependencies below that hold on the given entity-relationship diagram.

✓ $\{a_1\} \rightarrow \{a_2\}$
✓ $\{a_1\} \rightarrow \{a_3, a_4, a_5\}$
✓ $\{a_1\} \rightarrow \{a_6\}$
D. $\{a_2\} \rightarrow \{a_1\}$
E. $\{a_2\} \rightarrow \{a_3, a_4, a_5\}$

Select X if there is no answer.

Notes: $\{a_1\} \rightarrow \{R\}$ due to the (1, 1) cardinality with R_1.

17. (3 points) **(MRQ)** Select **ALL** the functional dependencies below that hold on the given entity-relationship diagram.

✓ $\{a_3, a_4\} \rightarrow \{a_1, a_2\}$
✓ $\{a_3, a_4\} \rightarrow \{a_5\}$
✓ $\{a_4, a_5\} \rightarrow \{a_1, a_2\}$
✓ $\{a_4, a_5\} \rightarrow \{a_3\}$
✓ $\{a_3, a_4, a_5\} \rightarrow \{a_1, a_2\}$

Select X if there is no answer.

Notes: $\{a_1\} \rightarrow \{R\}$ due to the (1, 1) cardinality with R_2.

18. (3 points) **(MRQ)** Select **ALL** the functional dependencies below that hold on the given entity-relationship diagram.

A. $\{a_3\} \rightarrow \{a_6\}$
B. $\{a_4\} \rightarrow \{a_6\}$
C. $\{a_5\} \rightarrow \{a_6\}$
✓ $\{a_3, a_4\} \rightarrow \{a_6\}$
✓ $\{a_4, a_5\} \rightarrow \{a_6\}$

Select X if there is no answer.

Notes: $\{a_3, a_4\} \rightarrow \{a_6\}$ because $\{a_3, a_4\} \rightarrow \{a_1\}$ and $\{a_1\} \rightarrow \{a_6\}$. Similar reasoning for $\{a_4, a_5\} \rightarrow \{a_6\}$.

19. (3 points) **(MRQ)** Select **ALL** the canonical cover of the set of functional dependencies that hold on the given entity-relationship diagram.

✓ $\{\ \{A\} \rightarrow \{D, E\},\ \{C, D\} \rightarrow \{E\},\ \{D, E\} \rightarrow \{A, B, C, F\}\ \}$
B. $\{\ \{A\} \rightarrow \{B, C, D, F\},\ \{C, D\} \rightarrow \{E\},\ \{D, E\} \rightarrow \{C\}\ \}$
✓ $\{\ \{A\} \rightarrow \{C, D\},\ \{C, D\} \rightarrow \{A, B, E, F\},\ \{D, E\} \rightarrow \{C\}\ \}$
D. $\{\ \{A\} \rightarrow \{D\},\ \{A\} \rightarrow \{E\},\ \{C, D\} \rightarrow \{E\},\ \{D, E\} \rightarrow \{A, B, C, F\}\ \}$
E. $\{\ \{A\} \rightarrow \{C\},\ \{A\} \rightarrow \{D\},\ \{C, D\} \rightarrow \{A, B, E, F\},\ \{D, E\} \rightarrow \{C\}\ \}$

Select X if there is no answer.

Notes: **C** and **D** are not canonical but minimal. **B** is missing $\{C, D\} \rightarrow \{A\}$ or $\{D, E\} \rightarrow \{A\}$. In general, we can simply check if they are (i) minimal then merge or (ii) they are equivalent.

20. (3 points) **(MRQ)** Let $R = \{a_1, a_2, a_3, a_4, a_5, a_6\}$ with Σ being the set of functional dependencies that hold on the given entity-relationship diagram. Select **ALL** the *superkeys* of R with Σ.

✓ $\{A, B\}$ B. $\{B, C\}$ ✓ $\{C, D\}$ ✓ $\{D, E\}$ ✓ $\{A, E\}$

Select X if there is no answer.

Notes: The keys are $\{A\}$, $\{C, D\}$, and $\{D, E\}$. So, the superset of these are the superkeys.

21. (3 points) **(MRQ)** Let $R = \{a_1, a_2, a_3, a_4, a_5, a_6\}$ with Σ being the set of functional dependencies that hold on the given entity-relationship diagram. Select **ALL** the *keys* of R with Σ.

A. $\{A, B\}$ B. $\{B, C\}$ ✓ $\{C, D\}$ ✓ $\{D, E\}$ E. $\{A, E\}$

Select X if there is no answer.

Notes: The keys are $\{A\}$, $\{C, D\}$, and $\{D, E\}$.

Continue on the next page...

Relational Algebra

For the next **three (3)** questions, we will consider the following schema.

```
CREATE TABLE country (
  code   VARCHAR(3)  PRIMARY KEY,
  name   VARCHAR(64)
);

CREATE TABLE airport (
  name     VARCHAR(5) PRIMARY KEY,
  country  VARCHAR(3) REFERENCES country(code)
);

CREATE TABLE flight (
  num      VARCHAR(7) PRIMARY KEY,
  takeoff  VARCHAR(5) REFERENCES airport(name),
  landing  VARCHAR(5) REFERENCES airport(name)
);
```

22. (3 points) **(MCQ)** Which of the following relational algebra queries finds all the different airport name (*i.e.,* `airport.name`) in `'Singapore'`.

    A. $\pi_{[a.name]}(\sigma_{[(c.\text{code}=a.\text{country})\wedge(c.\text{name}=\text{'Singapore'})]}(\rho(\text{country}, c) \times \rho(\text{airport}, a)))$

    B. $\pi_{[a.name]}(\sigma_{[(c.\text{code}=a.\text{country})]}(\rho(\text{country}, c) \bowtie_{[(c.\text{name}=\text{'Singapore'})]} \rho(\text{airport}, a)))$

    C. $\pi_{[a.name]}(\sigma_{[(c.\text{name}=\text{'Singapore'})]}(\rho(\text{country}, c)) \bowtie_{[(c.\text{code}=a.\text{country})]} \rho(\text{airport}, a))$

    ✓ *All of the above.*

    E. *None of the above.*

> **Notes:** These are all the same queries.

23. (3 points) **(MCQ)** Which of the following relational algebra queries finds all the different country code (*i.e.,* `country.code`) without any airport. Assume that `x ISNULL` check if `x` has a `NULL` value or not.

    A. $\pi_{[c.code]}(\rho(\text{country}, c) - \rho(\text{airport}, a))$

    B. $\pi_{[a.country]}(\rho(\text{airport}, a)) - \pi_{c.code}(\rho(\text{country}, c))$

    ✓ $\pi_{[c.code]}(\sigma_{a.\text{name}\ \texttt{ISNULL}}(\rho(\text{country}, c) ⟕_{[c.\text{code}=a.\text{country}]} \rho(\text{airport}, a)))$

    D. *All of the above.*

    E. *None of the above.*

> **Notes:** **A** is not *technically* union-compatible but PostgreSQL allows for some conversion. Still, the data will not match so this will give all country code. **B** is finding airport without country. **C** is equivalent to `EXCEPT` but we use outer join. The check with `ISNULL` ensures that we are only taking in the *dangling rows.*

24. (3 points) **(MCQ)** Which of the following relational algebra queries finds all the different flight number (*i.e.,* `flight.num`) that takeoff and landing in the same country.

   A. $\pi_{[f.num]}(\sigma_{[(a.\text{name}=f.\text{takeoff})\wedge(a.\text{name}=f.\text{landing})]}(\rho(\text{flight}, f) \times \rho(\text{airport}, a)))$

   B. $\pi_{[f.num]}(\sigma_{[(a.\text{name}=f.\text{takeoff})\wedge(a.\text{name}=f.\text{landing})\wedge(c.\text{code}=a.\text{country})]}(\rho(\text{flight}, f)\times\rho(\text{airport}, a)\times\rho(\text{country}, c)))$

   C. $\pi_{[f.num]}(\sigma_{[(a.\text{name}=f.\text{takeoff})\wedge(a.\text{name}=f.\text{landing})\wedge(a.\text{country}=a.\text{country})]}(\rho(\text{flight}, f) \times \rho(\text{airport}, a)))$

   D. *All of the above.*

   ✓ *None of the above.*

Notes: These are all taking off and landing in the same airport. Which is stricter than the same country. Note that because of the following two,

- $(a.\text{name} = f.\text{takeoff})$
- $(a.\text{name} = f.\text{landing})$

we can conclude the following.

- $(f.\text{landing} = f.\text{takeoff})$

This shows that the flight taking off and landing in the same airport.

Notes: For Q22 and Q23, you can try the following. Can you think about Q24?

```sql
CREATE TABLE country (
  code VARCHAR(3) PRIMARY KEY,
  name VARCHAR(64)
);
CREATE TABLE airport (
  name VARCHAR(5) PRIMARY KEY,
  country VARCHAR(3) REFERENCES country(code)
);
CREATE TABLE flight (
  num VARCHAR(7) PRIMARY KEY,
  takeoff VARCHAR(5) REFERENCES airport(name),
  landing VARCHAR(5) REFERENCES airport(name)
);

INSERT INTO country VALUES ('SGP', 'Singapore');
INSERT INTO country VALUES ('IDN', 'Indonesia');
INSERT INTO country VALUES ('MYS', 'Malaysia');
INSERT INTO country VALUES ('AND', 'Andorra');


INSERT INTO airport VALUES ('XSP', 'SGP');
INSERT INTO airport VALUES ('SIN', 'SGP');

INSERT INTO airport VALUES ('CGK', 'IDN');
INSERT INTO airport VALUES ('HLP', 'IDN');

INSERT INTO airport VALUES ('KUL', 'MYS');
INSERT INTO airport VALUES ('SZB', 'MYS');

-- Q22
SELECT a.name
FROM country c, airport a
WHERE c.code = a.country
  AND c.name = 'Singapore';
```

```sql
SELECT a.name
FROM country c JOIN airport a
     ON c.name = 'Singapore'
WHERE c.code = a.country;

SELECT a.name
FROM (SELECT * FROM country c WHERE c.name = 'Singapore') as c
     JOIN airport a ON c.code = a.country;

-- Q23
SELECT c.code
FROM (
  SELECT * FROM country c
  EXCEPT
  SELECT * FROM airport a
) as c;

SELECT a.country
FROM airport a
EXCEPT
SELECT c.code
FROM country c;

SELECT c.code
FROM country c LEFT JOIN airport a
     ON c.code = a.country
WHERE a.name ISNULL;
```

Continue on the next page...

Miscellaneous

25. (11 points) Consider the following relation and set of functional dependencies. Note that the set of functional dependencies is already given in canonical cover.

$$R = \{A, B, C, D, E\}$$
$$\Sigma = \{\ \{A, B\} \rightarrow \{C, D\},\ \{C\} \rightarrow \{A, B\},\ \{D\} \rightarrow \{E\}\ \}$$

(a) (2 points) Find all the *candidate keys* (*i.e.,* keys) of R with Σ.

> **Notes:** $\{\ \{C\}, \{A, B\}\ \}$

(b) (2 points) Find a functional dependency that violates 3NF property R with Σ.

> **Notes:** $\{D\} \rightarrow \{E\}$

(c) (4 points) Since R is not in 3NF, we want to make it into 3NF **but not** BCNF. We want to do this by adding one **and only one** functional dependency to Σ.
Let us call this new set functional dependencies Σ'. Ensure that R is in 3NF but not in BCNF given Σ'.

> **Notes:** $\{E\} \rightarrow \{A\}$ or $\{E\} \rightarrow \{B\}$.
>
> The idea is to make E to be a prime attribute without making D a candidate key. If we add $\{E\} \rightarrow \{A, B\}$ or $\{E\} \rightarrow \{C\}$ then D will be a candidate key.

(d) (3 points) Find all the *candidate keys* (*i.e.,* keys) of R with Σ'.

> **Notes:**
>
> - If $\{E\} \rightarrow \{A\}$ then $\{\ \{C\},\ \{A, B\},\ \{B, D\},\ \{B, E\}\ \}$.
> - If $\{E\} \rightarrow \{B\}$ then $\{\ \{C\},\ \{A, B\},\ \{A, D\},\ \{A, E\}\ \}$.

> **Notes:** Notice that we are talking about the entire schema R. Even when there is no overlap in the keys, the schema R may still violate BCNF and 3NF.

This page is intentionally left blank.

END OF PAPER